**PCT** WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 : G11B 20/18

A1

(11) International Publication Number: **WO 99/48097**

(43) International Publication Date: 23 September 1999 (23.09.99)

(21) International Application Number: PCT/IT98/00056

(22) International Filing Date: 18 March 1998 (18.03.98)

(71) Applicant *(for all designated States except US)*: STMICROELECTRONICS S.R.L. [IT/IT]; Via C. Olivetti, 2, I–20041 Agrate Brianza (IT).

(72) Inventors; and

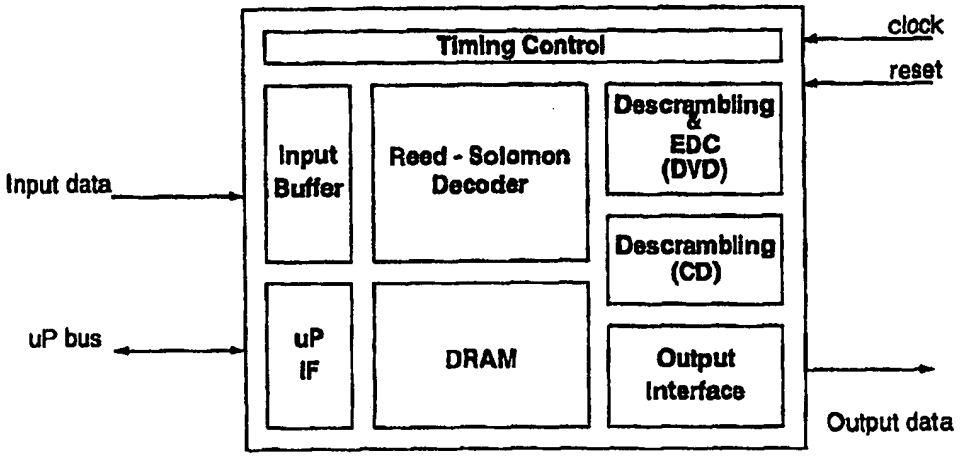
(75) Inventors/Applicants *(for US only)*: DE MARZI, Giuseppe [IT/IT]; Viale Monte Santo, 8, I–20124 Milano (IT). ANDOLINA, Giancarlo [IT/IT]; Via Pio X, 9, I–20049 Concorezzo (IT). UGIOLI, Roberto [IT/IT]; Via Cesare Battisti, 21/D, I–20037 Paderno Dugnano (IT). BRENNA, Filippo [IT/IT]; Via Don Zonca, 17, I–20050 Besana Brianza (IT).

(74) Agent: PELLEGRI, Alberto; Società Italiana Brevetti S.p.A., Via Puccini, 7, I–21100 Varese (IT).

(81) Designated States: JP, US, European patent (AT, BE, CH, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE).

Published
*With international search report.*

(54) Title: REED–SOLOMON DECODING OF DATA READ FROM DVD OR CD SUPPORTS




(57) Abstract

An effective organization and the transferring data among the functional blocks of an integrated system (ECC–IC) of a read channel of data recorded on DVD–Rom, DVD–Ram, DVD–R or CD Rom for performing Reed–Solomon decoding including off–line heroic correction, or deinterleaving, Reed–Solomon decoding and correction, said integrated system including an input buffer (INPUT BUFFER), an interface with a microcontroller bus ($\mu$P IF), a Reed–Solomon decoder (REED–SOLOMON DECODER), an embedded RAM (DRAM) accessed through a 17–bit bus, a descrambling and EDC control block (DESCRAMBLING & EDC) for DVD modes of operation a descrambling block (DESCRAMBLING CD) for CD modes of operations, a data output interface (OUTPUT INTERFACE) and a timing control block (TIMING CONTROL), permits the decoding of the input data acquired through said input buffer (INPUT BUFFER) at a rate of up to four–times the reference bit rate of DVD formatted data using a clock for accessing said embedded RAM having a frequency half that of the clock that is used in said Reed–Solomon decoder, while reducing the number of accesses to said embedded RAM needed to perform the decoding.

*FOR THE PURPOSES OF INFORMATION ONLY*

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| AL | Albania | ES | Spain | LS | Lesotho | SI | Slovenia |
| AM | Armenia | FI | Finland | LT | Lithuania | SK | Slovakia |
| AT | Austria | FR | France | LU | Luxembourg | SN | Senegal |
| AU | Australia | GA | Gabon | LV | Latvia | SZ | Swaziland |
| AZ | Azerbaijan | GB | United Kingdom | MC | Monaco | TD | Chad |
| BA | Bosnia and Herzegovina | GE | Georgia | MD | Republic of Moldova | TG | Togo |
| BB | Barbados | GH | Ghana | MG | Madagascar | TJ | Tajikistan |
| BE | Belgium | GN | Guinea | MK | The former Yugoslav Republic of Macedonia | TM | Turkmenistan |
| BF | Burkina Faso | GR | Greece | | | TR | Turkey |
| BG | Bulgaria | HU | Hungary | ML | Mali | TT | Trinidad and Tobago |
| BJ | Benin | IE | Ireland | MN | Mongolia | UA | Ukraine |
| BR | Brazil | IL | Israel | MR | Mauritania | UG | Uganda |
| BY | Belarus | IS | Iceland | MW | Malawi | US | United States of America |
| CA | Canada | IT | Italy | MX | Mexico | UZ | Uzbekistan |
| CF | Central African Republic | JP | Japan | NE | Niger | VN | Viet Nam |
| CG | Congo | KE | Kenya | NL | Netherlands | YU | Yugoslavia |
| CH | Switzerland | KG | Kyrgyzstan | NO | Norway | ZW | Zimbabwe |
| CI | Côte d'Ivoire | KP | Democratic People's Republic of Korea | NZ | New Zealand | | |
| CM | Cameroon | | | PL | Poland | | |
| CN | China | KR | Republic of Korea | PT | Portugal | | |
| CU | Cuba | KZ | Kazakstan | RO | Romania | | |
| CZ | Czech Republic | LC | Saint Lucia | RU | Russian Federation | | |
| DE | Germany | LI | Liechtenstein | SD | Sudan | | |
| DK | Denmark | LK | Sri Lanka | SE | Sweden | | |
| EE | Estonia | LR | Liberia | SG | Singapore | | |

# REED-SOLOMON DECODING OF DATA READ FROM DVD OR CD SUPPORTS

## FIELD OF THE INVENTION

The present invention relates to a read-channel circuitry of data read from a mass memory support such as a DVD or a CD and more in particular to an integrated system for decoding according to the Reed-Solomon algorithm data read from a mass memory support coded according to standard DVD-ROM, DVD-R, DVD-RAM or CD-ROM protocols.

## BACKGROUND

DVD and CD optical supports are more and more used for storing large quantities of data in PC's, digital audio and video playback systems and the like. The storing and reading of data to and from these supports imply the coding and decoding of data according to standard protocols that are defined at international level (e.g. ISO/IEC, CEI/IEC, etc.).

In write-read channel circuitry, reliability in terms of ability of detecting and correcting errors, especially during a phase of decoding of the coded data read from the support during a reading phase, and speed are of paramount importance. Obvious cost-effectiveness considerations call for the highest level of integration of system and/or subsystem circuitries in the minimum number of distinct integrated circuits. Multifunctional Reed-Solomon decoders, capable of handling either DVD decoding and correction or CIRC decoding and correction for all the commonly used CD-modes should be advantageously integrated in a single device including an embedded RAM required for the decoding and correction operations on a bitstream of input data as produced by the data aquisition means of the read channel.

An architectural layout of such a multifunctional decoder ECC-IC is depicted in Fig. 1.

The integrated decoder handles CD modes bitstreams of any format as well as DVD-ROM, DVD-RAM and DVD-R mode bitstreams and advantageously should possess speed capabilities of handling bitstreams equivalent to a significantly large multiple of standard or base CD rates and of standard or base DVD rates.

With reference to the functional diagram of Fig. 1, the multifunctional integrated decoder ECC-IC, when operating in DVD mode, performs horizontal and vertical decoding of the input data stream and the decoded data are then descrambled and EDC checked before sending them to an output interface circuitry. When operating in a CD mode, the data are decoded and deinterleaved without performing any C3 decoding. Finally, when functioning in a BCA mode, the integrated decoder may perform a Burst Cutting Area decoding of the data stream.

The input data stream consists of the signals output by a data aquisition IC as depicted in Fig. 2. The signals contain data, information about the data and address information.

**byte_clk** [1]

The byte_clk (byte clock) signal indicates that the data byte can be read. It is generated once per data byte for 1 system clock cycle.

**erasure [1]**

The erasure bit is set to 1 if the current data byte is not a valid 16/8 modulation pattern (14/8 for CD) -- if the pattern is valid, erasure is set to 0.

**data [8]**

The 8-bit data bus contains the demodulated data byte.

**SID [4]**

The 4-bit SID (sector ID) contains the 4 least significant bits of the logical sector ID. This signal provides the sector address within each block.

**id_error [1]**

An id_error bit of 0 indicates that the SID was decoded with no errors and no corrections. If the SID contained errors (no correction possible) or if a single error was corrected, the id_error bit is set to 1.

**DVD/BCA frame_address[4:0] or CD S0/S1**

In DVD modes the acquisition part keeps a memory of the syncs received and from this history extract the 5-bit frame address.

In BCA mode frame_address[3:0] depends from the sync found ($SB_{BCA}$, $RS_{BCA}1$, ..., $RS_{BCA}n$, $RS_{BCA}13$, $RS_{BCA}14$, $RS_{BCA}15$).

In CD mode the S0 and S1 signals are sent on frame_address[0] and frame_address[1] respectively.

**CD nxfr [1] or DVD next_frame[1]**

The DVD next_frame indicates that a new DVD frame is starting. The CD nxfr signal indicates that a new CD frame is starting. The BCA next_frame indicates that a BCA Re-sync has been found.

The timing diagrams of the input data aquisition for the case of operation in DVD mode and in CD mode are shown in figures 3 and 4, respectively.

ECC-IC has two kinds of output interface: one for CD-modes (serial) and one for DVD-modes (parallel). In particular, the CD output interface may be a common $I^2$

S interface employing a format as depicted in Fig. 5, and the subcode interface has as format as depicted in Fig. 6.

The Reed-Solomon decoder block depicted in Fig. 7, supports five main modes:

| | |
|---|---|
| 1) DVD Outer code (208,192,17) | 8 errors or 16 erasures |
| 2) DVD Inner code (182, 172, 11) | 5 errors or 10 erasures |
| 3) CD C1 code (32,28,5) | 2 errors |
| 4) CD C2 code (28,24,5) | 2 errors or 4 erasures |
| 5) BCA code (52,48,5) | 2 errors or 4 erasures |

The number of erasures that can be corrected is programmable, depending on the mode, from 13 up to 16 for DVD Outer, from 7 up to 10 for DVD Inner, from 1 up to 4 for CD C2. The Reed-Solomon block can be programmed to make a severe check for "miscorrections": This preselection will cause a reduction of the decoding performance.

## THE ERROR CORRECTION ALGORITHM

The complete algebraic decoding algorithm for the errors and the erasures is summarized in the following steps:

STEP 1. Calculation of the syndrome S(z), the erasure locator polynomial E(z) and the calculation of the modified syndrome T(z). If r(x) is the received code word

$$S_j = \sum_{i=0}^{n-1} r_{n-1} \alpha^{ji}$$

If $\alpha^{jk}$ is the position of a k-erasure and e is the number of erasures

$$E(z) = \prod_{k=1}^{e} (1 - z\alpha^{j_k})$$

If t is the maximum number of errors the code is able to correct

$$T(z) = S(z)E(z) \bmod (z^{2t})$$

STEP 2. Perform the extended Euclidean Algorithm (modified version) to calculate the error locator polynomial $\sigma(z)$ and the error evaluator polynomial $\omega(z)$. Calculation of the new error locator polynomial $\Psi(z)$

$$\Psi(z) = \sigma(z)E(z)$$

STEP 3. Perform the Chien search to find the roots of the new error locator polynomial. The roots of this polynomial indicate the error and erasure positions in a received code word. Perform the Forney's algorithm to find the error and erasure values.

STEP 4. Check the decoding process and correct the received code word.

The timing control block sets the control inputs of the Reed-Solomon block and send start syndrome pulse (*start_synd*) with the first symbol of the code word.

The *en_synd* is acting as an enable for the data bus (data_in).

Every erasure should be flagged by *erasure_pos*. During the shifting of the code word, the Reed-Solomon calculates its syndrome and its erasure polynomial.

Once the code word is completely shifted into the Reed-Solomon, the controller has to start the Key Equation Solver (*start_kes*).

The Reed-Solomon responds when error and location values are ready for the controller (*kesready*).

The errors and error locations are stored in a Lifo and the controller can read them with *read_pos* and *read_error* signals.

The Reed Solomon processing consists of three main tasks:

a) syndrome and erasure polynomial calculation (invoked by *start_synd*)

b) key equation solving and error calculation (invoked by *start_kes*)

c) Chien & Forney (generating the ending signal *kes_ready*)

as depicted in the block diagram of Fig. 8.

In DVD Outer decoding, the erasure polynomial is equal for each column, because it is calculated using the incorrectable flag coming from the Inner decoding. For this reason it is calculated only once for each Ecc block of data at the beginning of the vertical decoding process; the resulting polynomial is stored depending on the *store_eras_poly* signal. This is depicted by way of a block diagram in Fig. 9.

During the DVD decoding process (Inner - Outer - Inner - Outer - .. ) the signal *sel_eras_poly* send to Key Equation Solver the current erasure polynomial (Inner) or the previous stored erasure polynomial (Outer).

## SCOPE AND SUMMARY OF THE INVENTION

To overcome the intrinsic low speed of embedded DRAMs it has been found that a considerable increase of the required processing speed capabilities in a fully integrated decoder (ECC-IC) may be obtained by organizing the data flow within the integrated decoder and the embedded RAM in a way as to **reduce the number of accesses** to the embedded RAM needed to perfom the decoding at the

required speed while using the Reed-Solomon decoding block at twice the maximum clock frequency allowed by the embedded RAM (25MHz).

Accordingly, in DVD decoding, the syndrome engine is made capable to process two code words at the same time, by storing the final polynomials into distinct registers: two for storing vertical syndromes and one for storing horizontal syndromes, thus implementing a kind of parallel processing in the decoding phase.

Accordingly, in CD decoding, a new addressing scheme has been found to perform very high speed M2 deinterleaving, minimizing the number of accesses to the embedded DRAM required to perform the deinterleaving of data.

The embedded DRAM is organized in distinct banks, each divided into a number of pages of certain capacity of words, functioning in a synchronous page mode.

The method and architecture of the multifunctional decoder of the invention are defined in the appended claims.

## BRIEF DESCRIPTION OF THE DRAWINGS

Fig. 1 is an architectural layout of the multifunctional decoder ECC-IC of the invention;

Fig. 2 shows the input data coming from a data acquisition IC;

Fig. 3 and 4 show the timing diagram of the input data acquisition for the case of operation in DVD mode and in CD mode, respectively;

Fig. 5 and 6 illustrate the output serial $I^2S$ data interface for CD mode and the output serial subcode interface for CD mode, respectively;

Fig. 7 represents the Reed-Solomon Decoder block;

Fig. 8 schematically shows the processing performed by the Reed-Solomon Decoder;

Fig. 9 schematically illustrates the processing of erasure flags during horizontal and vertical DVD decoding;

Fig. 10 shows the parallel processing of partial syndrome values;

Fig 11 shows the timing diagram of the embedded DRAM working in a synchronous page mode, for a word access in first cycle;

Fig. 12 shows the functional blocks of the ECC-IC device that are involved with the data flow in a DVD mode of operation;

Fig. 13 illustrates schematically the different phases of processing (DVD mode);

Fig. 14 shows the particular mapping adopted for the embedded DRAM for one ECC-block of data;

Fig. 15 illustrates the output processing;

Fig 16 illustrates the processing referred to a complete elaboration time frame, divided in four steps;

Fig. 17 shows the functional blocks of the ECC-IC device that are involved with the data-flow for the case of a CD mode of operation;

Fig. 18 illustrates the first step of the deinterleaving (M1-deinterleaving) of input data;

Fig. 19 illustrates the particular mapping inside a page of the embedded DRAM (M2-deinterleaving);

Fig. 20 shows deinterleaved frames read from the DRAM;

Table 1 illustrates the particular mapping of the two banks of DRAM (M2-deinterleaving) refering to the input frame numbers.

Fig. 21 depicts the enable signal according to the different frame numbers;

Fig. 22 shows the particular mapping inside a page of the embedded DRAM (M3-deinterleaving);

Fig. 23 illustrates schematically the different phases of processing (CD mode);

Fig. 24 illustrates how one block of 12 words is read from memory for each input frame;

Fig. 25 shows the functional blocks of the ECC-IC device that are involved with the data flow during operation in BCA mode;

Fig. 26 illustrates the mapping of the embedded DRAM (BCA mode);

Fig. 27 illustrates schematically the different phases of processing (BCA mode)

Fig. 28 depicts the descrambling architecture for DVD mode of operation;

Fig. 29 depicts the descrambling architecture for CD mode of operation;

Fig. 30 shows the functional block diagram of the EDC checker;

Fig. 31 depicts the way the new CRC value is produced by shifting the old value and xoring the result with the updating value.

## DESCRIPTION OF AN EMBODIMENT OF THE INVENTION

An integrated multifunctional Reed-Solomon decoder according to the general characteristics already described above and embodying the features of the invention, implementing the method of the invention of organising and transferring data among the functional blocks of the integrated decoder will now be described with reference to the drawings.

With reference to the diagram of Fig. 10, the *synd2* value (rs_reg[2]), if set, selects for each byte_clk a different syndrome register (SK1, SK2) to store the polynomials related to the two different columns. The *sel_synd_poly* signal chooses which one of the stored syndrome values is to be sent to Key Equation Solver (KES).

### *TIMING CONTROL BLOCK*

According to the sample embodiment taken into consideration, the following assumptions were made:

- The system clock speed is 50MHz.
- The symbols coming from the data acquisition block are stored into an input memory working with the system 50MHz clock.
- The embedded DRAM works in a synchronous page mode with a 25MHz clock.
- DVDx4 and CDx32 are taken as target speed capabilities for the relative modes (two steps of decoding for DVD modes and complete CIRC decoding for CD modes).
- The byte clock coming from the data acquisition device is always available.

- At the output of ECC-IC only complete Ecc blocks of data will be output.

The characteristics of the DRAM embedded in the ECC-IC chip were the following:

- 1-Transistor Planar memory cell.
- Planar Memory cell area: 11.2 mm2.
- Organized in three banks of 16K words of 17 bits, split into 512 rows of 32 words of 17 bits.
- Byte-Write capability available, i.e. write operation can be applied to any of the two bytes. Read operation is always done on 17 bits.
- 512-cycle refresh distributed across 1ms.
- No Redundancy
- Latched Output Data.
- Bidirectional data bus.
- On-chip voltage generators
- Separate I/O data buses with write-through enable pin.

During page modes, row addresses are latched for the duration of whole row access, i.e until precharge is performed, while column addresses are latched only during the actual column access.

Row is accessed first, then words in that row (or page) at a rate of one word per cycle. An extra cycle is required to go back to precharge before accessing a new row.

There is no specific Refresh cycle. Any Read or Write operation is also a refresh cycle.

The timing diagram of the embedded DRAM working in a synchronous page mode, for a word access in first cycle, is shown in Fig. 11.

The Timing Control block is responsible of scheduling data transfer among the following functional blocks that compose the ECC-IC of Fig. 1:

- Input interface
- Input memory
- Reed Solomon decoder
- DRAM (embedded in the IC)
- Descrambling and EDC checker (for DVD mode of operation only)
- Descrambling (for CD modes of operation only)
- Output interface

*DVD MODE OF OPERATION*

Processing of data starts each time a couple of rows has been received.

The decoder ECC-IC performs the following operations:

- horizontal correction of 2 rows, Ecc-block(#n).
- vertical correction of 2 columns, Ecc-block(#n-1) (this operation starts only when the horizontal correction of this Ecc-block has been completed).
- output of 2 rows, Ecc-block(#n-1) or Ecc-block(#n-2) (this operation starts when the vertical correction of the Ecc-block has been completed).

In order to support a sequential reading, the ECC-IC must complete the above processing before a new couple of rows is received.

The minimum time frame when operating in a DVDx4 mode is 56.875usec (2x182/6.4MHz).

Fig. 12 shows the functional blocks of the ECC-IC device that are involved by the data flow in a DVD mode of operation.

*Normal mode*

Data and erasure flag coming from the data acquisition IC are sampled into Input interface and stored frame after frame into one of the two Input RAMs.

Each time 4 frames (equivalent to two rows) have been received, the input is switched on the second RAM; the previous rows are read from this memory and sent both to the Reed-Solomon decoding block, to calculate Inner syndromes and corresponding erasure polynomial, and to the embedded DRAM, where two complete ECC blocks of data are stored.

ECC-IC performs at the same time the Inner decoding of Ecc block #n and the Vertical decoding of Ecc block #(n-1).

. Flags coming from Inner decoding are stored into DRAM to be used later in Outer decoding: for $Row_n$, the erasure flag is stored into the ($Row_n$, $Col_{182}$) word. Horizontal corrections are effected (up to 5 errors or 10 erasures) within the DRAM.

While storing rows into one Ecc block in the DRAM, columns of the previous Ecc block are being decoded. Each time a word is read from DRAM, bytes of two different columns are retrieved, therefore the syndrome engine is able to calculate syndromes of both columns at the same time.

Vertical corrections are effected (up to 8 errors or 16 erasures) within the DRAM.

When an Ecc block is completely decoded vertically, the output processing begins: data are read by rows from DRAM and sent to the descrambler block, and successively to the EDC checking block and finally to the output interface block.

Each time EDC fails, a bit is set into a register (16 bits), containing the results of each sector. At the first fail an interrupt is generated towards the μP.

Cycles are inserted to guarantee complete refreshing of the embedded DRAM.

The different phases of processing are schematized in Fig. 13.

Each Elab frame starts only when two complete rows have been received and are ready to be horizontally decoded and stored into DRAM. For this reason the byte clock should never disappear, in order to avoid arresting the internal processing.

In the time frame two new rows are received, two rows and two columns are decoded. Therefore, the Elab time frame must be shorter than twice the shortest Input data time frame (DVDx4) = 2*(28.43 usec).

In view of this length of each Elab frame, the decoding core will be always waiting for the rows_ready signal. If for any reason (as for example an unlocked DPLL) the Input data frame will be shorter than Elab frame the latest stored row will be overwritten in the Input RAM by data coming from the data acquisition IC.

With a DRAM functionning in synchronous page mode, each time a new page has to be accessed, one extra cycle is needed to perform the pre-charging of the page.In one page of DRAM,, 16 bytes per row and 12 bytes per column are stored: 1row = 12 pages, 1 column = 18 pages. To store 2 Ecc blocks 432 pages are necessary. 32 rows are refreshed during each Elab frame.

At the end of each Elab frame, the core waits for the next two rows to decode, while refreshing the DRAM. As soon as a new couple of rows is received ready to be decoded, the process restarts.

*Heroic Correction*

If for any reason the output data are still affected by errors, the μP may decide to initiate a more effective decoding of a specific Ecc block of data.

The μP can decide how many iterations of decoding should be done (Inner - Outer - Inner - Outer - .....- Inner - Outer); the maximum number of ECC-blocks is written in a ECC_block register, the number of iteration is written into a nb_iter register and the process is started setting another bit in the same register (HC).

Data coming from the data acquisition IC are horizontally and vertically corrected as in normal mode. The flags generated from the vertical decoding are stored into DRAM to be used as erasure flags in the horizontal decoding step of the selected Ecc-block.

At the end of the vertical decoding the ECC-block number is compared with the value written in the Ecc_block register. Ifthe ECC-block number matches, the input is disabled and the next step of decoding takes place on this ECC-block: for $Col_n$ the erasure flag is stored into the ($Row_{208}$, $Col_n$) word.

Data are read towards the DESCRAMBLING, EDC AND OUTPUT INTERFACE blocks and, at the same time, horizontal syndromes are calculated and corrections applied within the DRAM.

At the end of the Ecc block, if any of EDC check bit is set and if the maximum number of iterations is not reached the process go ahead with another vertical decoding step and so on.

When all the EDC checks are right or when the maximum number of iterations has been reached, an interrupt is generated and the processing is stopped.

The algorithm may be expressed as follows:

STEP1: read Ecc blocks from disk. Normal decoding of data with check, after vertical step, of the Ecc-block number. No output of data during this step. This phase is repeated until the right Ecc-block number will be found. When this happens, the input of following data will be disabled.

STEP2: Output data + EDC check + Inner correction (use flag coming from outer decoding as erasure flag).

if (EDC is OK) OR (maximum number of iterations is reached)

STOP

else

vertical correction

repeat this step

*Long Mode*

If a specific bit is set into a dedicated register (longm), the ECC-IC may work in long mode: the decoding process is still active but no data correction is effected and all bytes received from acquisition (user bytes + parity bytes) are sent to the output interface.

*Input RAM*

Two static Ram, each one 2 rows wide (2x182x9bits), are employed.

While data are stored into one RAM frame after frame, the previous two rows are read from the other RAM. Only when two rows are completely built, the transfer to Reed-Solomon decoder and to the embedded DRAM begins.

For each location 1 byte and its erasure flag coming from the demodulation process are stored.

*Embedded DRAM*

3 banks of 16Kx17 bits of DRAM memory are employed. Each bank is split into 512 rows of 32 words of 17 bits.

In DVD mode the three banks are addressed as a single memory; each ECC block is stored according to the following mapping: in each page (3x32x17bits) 16 bytes of one row and 12 bytes of one column are stored.

To store or to retrieve a complete row, 12 pages must be addressed, to read a complete column, 18 pages must be addressed.

In this mode, the 17th bit of each location becomes meaningless.

An extra-row ($Row_{208}$) and an extra-column ($Col_{182}$) are needed to store erasure flags coming from Inner and Outer decoding.

According to an important aspect of this invention, an effective mapping becomes intrumental in reducing as much as possible the number of accesses to the DRAM.

By operating in a synchronous page mode, a cycle is lost at each change of page, because a precharge cycle is normally needed.

By contrast, if when changing page the bank of DRAM is also changed, no extra cycle is needed. In practice, in writing a row, the page and also the DRAM bank are changed: every 16 bytes (bank0 - bank1 - bank2 - bank0 - ... and so on).

Therefore, in retrieving a column from the embedded DRAM during the decoding, it is always true that a page change occurs only when changing bank (from bank2 to bank0), thus there are no extra cycles.

Extra cycles are needed only during horizontal and vertical correction, because in those operations, it may happen to be changing page while remaining in the same bank of the embedded DRAM.

Of course, also during refresh there occurs an extra cycle, because all the banks are refreshed at the same time (dummy read).

The particular mapping of the embedded DRAM for one Ecc_block of data is depicted in Fig. 14.

To store one Ecc_block 216 pages are needed. Refresh is effected only for 432 pages of the DRAM because two Ecc_blocks of data are needed for the processing.

*Output interface - DVD modes*

Decoded sectors are read from DRAM row after row, descrambled and EDC checked if needed. The pages are read from memory (max 16 bytes, min 6 bytes) and stored into the output buffer (24 bytes).

When SREQI is asserted by an ATAPI-IC, data available in this buffer are read and sent to the output.

During the processing of input data the Timing Control samples the status bit of the output buffer to be ready to store another page of decoded data read from the DRAM as soon as enough data are sent to the ATAPI-IC.

Until the Elab frame is finished, 2 rows as maximum can be trasmitted to the output buffer. Instead at the end of internal processing and during refresh cycles, as many data as required by the ATAPI-IC are sent to the output buffer. For this reason decoded ECC blocks are sent to the ATAPI-IC always at the maximum speed allowed by SREQI signal.

The output process is depicted in Fig. 15.

*Parallel Vertical Syndromes Computation*

During vertical decoding, data are retrieved from DRAM word after word; in each word a byte belonging to one column and another belonging to the next one are identified. The Reed-Solomon decoder decodes at the same time data of the input rows and data of these columns.

The processing is depicted in Fig. 16. The Elab frame (computation time frame) may be divided in four steps:

STEP1.

- Horizontal syndrome computation (row#m, Eccb#n)

- Horizontal erasure polynomial computation (row#m, Eccb#n)

- Vertical syndrome decoding (col#p, Eccb#n-1)

STEP2.

- Horizontal syndrome computation (row#m+1, Eccb#n)

- Horizontal erasure polynomial computation (row#m+1, Eccb#n)

- Horizontal syndrome decoding (row#m, Eccb#n)

STEP3.

- Vertical syndromes computation (col#p+2,p+3, Eccb#n-1) (to be continued)

- Vertical syndrome decoding (col#p+1, Eccb#n-1)

STEP4.

- Vertical syndromes computation (col#p+2,p+3, Eccb#n-1)

- Horizontal syndrome decoding (row#m+1, Eccb#n)

During step3 and step4, the first column of the Eccb#n-1 to be read from DRAM is $Col_{182}$, containing the erasure flags from horizontal decoding from which the vertical erasure polynomial is updated and stored too.

*CD MODES OF OPERATION*

The functional circuit blocks of the ECC-IC involved into CD data flow are shown in Fig. 17.

In CD-modes the 3 banks of the embedded DRAM are used as three different memories: M2-deinterleaving is carried out in two of the banks, while M3-deinterleaving is performed in the third bank.

Data received from the data acquisition IC is stored into Input RAM (user data) or sent to output interface (subcode data and sync). For each F3 frame 32 bytes of user data are stored, each one with its erasure flag.

The first step of the deinterleaving (M1-deinteleaving) takes place into the Input RAM according to the sheme of Fig. 18.

Data are read from this memory deinterleaved, sent to Reed-Solomon decoder and stored into the correction FIFO. In this step the Reed-Solomon core calculates the C1 syndrome of the input frame.

Meanwhile, the previous frame stored into the FIFO is corrected on the fly and stored into the two banks (DRAM1, DRAM2) of the DRAM, organized in 512 pages of 32*17 bits each, according to a mapping as schematized in Fig. 19: the 28 bytes frames are stored in one DRAM page together with its C1 decoding flag. In a page of one bank of DRAM two 28 bytes-frame can be stored. The M2-deinterleaving takes place in these two banks of DRAM.

According to an important aspect of this invenction, an effective addressing of the incoming frames together with the mapping of Fig. 19 becomes instrumental in reducing as much as possible the number of accesses of the DRAM, allowing to perform a very high speed M2-deinterleaving.

112 frame need to be stored in the two banks, requiring 28 pages of DRAM1 and DRAM2: therefore only these pages need to be refreshed.

Tab. 1 shows the content of the two banks (DRAM1, DRAM2) after 112 frames have been received. Depending on the circular approach chosen, frame#113 and frame #121 will be stored in page#0 of DRAM1 and so on.

Functioning in a synchronous page mode, the embedded DRAM needs a cycle for precharge each time a new page is addressed.

Frames coming from the correction FIFO are corrected on the fly and stored in DRAM1 or into DRAM2: 32 corrected bytes are read from the FIFO but only 28 bytes are stored into DRAM, because the parity bytes can be discarded.

The operations performed when a frame is written into the right DRAM bank are:

- select the right bank and the right page

- write 14 words (17 bits) into the memory

To write a complete frame we need (1 + 14)cycles.

Data for C2 syndrome calculation (28 bytes + flags) are read from M2 memory: the new mapping of the data allows to reduce the number of accesses of the DRAM needed to retrieve both deinterleaved data and correspondent flags.

Fig. 20 shows that to retrieve a complete deinterleaved frame (1+2)*8=24 DRAM cycles are needed. In fact, to read a deinterleaved frame, 8 pages are addressed and 32 bytes retrieved: only 28 bytes are significant.

Firstly, the page where to start from must be decided, according to

$$firstPAGE = 4\left\lfloor \frac{frame}{16} \right\rfloor \bmod(28) + frame \bmod(4)$$

where "frame" stands for the frame number.

From each selected page 4 bytes in 2 cycles are read, two from DRAM1, two from DRAM2: for example, if page#0 is selected, in the first cycle a byte from frame#1 (DRAM1) and one from frame #5 (DRAM2) are read, in the second cycle a byte from frame #3 (DRAM1) and one from frame #9 (DRAM2) are read.

Every time 1 bytes from a page are retrieved, the page number is incremented according to

$$nextPAGE = (PAGE + 4) \bmod (28)$$

At this point, from the 32 bytes read, the 28 significant bytes are chosen: they are always consecutive but the first significant byte change position each 4 frames, still remaining in the first selected page.

$$firstBYTE = \left\lfloor \frac{frame \bmod (16)}{4} \right\rfloor$$

The enable signal is shown in Fig. 21, according to the different frame numbers. The waveform of this signal repeats itself every 16 frames.

M2-deinterleaved data are read from DRAM, sent to the Reed-Solomon decoder and stored into the correction Fifo again. In this step the Reed-Solomon core calculates the C2 syndrome.

The previous frame stored into the Fifo is corrected "on the fly" and stored into the M3 memory bank. Deinterleaved data are read from the M3 bank and sent to the output interface.

The third step of deinterleaving is done into the third bank (M3) of DRAM, organized in 512 pages of 32*17 bits each, by using only two pages for the process.

To perform this third step of deinterleaving four frames coming from C2 decoding are stored, each byte coming with its flag of correct/incorrect decoding.

First of all, two bytes are packed in one word ready to be written into M3, therefore for each couple of bytes only 1 flag in 17th position is written.

The mapping of the M3 DRAM memory is shown in Fig. 22.

The operations performed when a frame is written into the M3 memory are:

- select the right page

- write 12 words (17 bits) into the memory

To write a complete frame we need (1 + 12) cycles

The operations performed when a frame is read from the M3 memory are:

- select the right page

- read 12 words (17 bits) from the memory, two words from frame (N) then two words from frame (N-2)

To read a complete frame we need (1 + 12) cycles.

The Elab_frame is illustrated in Fig. 23.

The elaboration time frame must be shorter than the shortest Input time frame (for CDx32 rate it is of 4.25 usec).

The Elab frame start only if there is a new F2-frame to be decoded.

The C1-syndromes update time depends from the speed of the disk, because the Input memory is shared between the input process and the CIRC decoding.

The refresh of twenty eight pages is done at the same time on the three banks of the DRAM.

At the end of an Elab frame refresh cycles are inserted until a new F2-frame is ready for decoding.

*Output of data in CD modes*

Data are read from the third bank of the DRAM (M3), frame after frame, and are sent to the CD-descrambler (if enabled) and to output interface.

One blocks of 12 words is read from memory for each input frame as illustrated in Fig. 24. Each byte read from DRAM is sent to the output buffer.

*BCA MODE OPERATION*

BCA is an area for recording information after finishing disc manufacturing process. The data in the BCA-Code consists of a BCA-preamble field, a BCA-Data field and a BCA-Postamble field. The length of BCA-Data field is (16n-4) bytes, with $1 \leq n \leq 12$. These bytes are encoded with a Reed-Solomon code (52,48,5) and protected by a 32 bits CRC.

Fig. 25 shows the functional blocks of the ECC-IC device that are involved by the data flow in a BCA mode of operation.

A SRAM (364x9bits) is used for storing the incoming data. The sync information is carried by frame_addr[4:0].

Data are stored into Input RAM to reconstruct the complete block. When the last sync ($RS_{BCA15}$) or the postamble pattern is received, the decoding process starts.

Bytes are sent to the embedded DRAM and to the Reed-Solomon decoder to calculate the syndromes of the four code words. During this transfer, zeroes are inserted to fit the code word length (if required).All the bytes (information + parity bytes = 4*(48+4)bytes) are stored into DRAM, following the same addressing scheme as in DVD-modes: code word 0 is equivalent to Row0, code word 1 to Row1, code word 2 to Row2, code word 3 to Row3. The mapping is illustrated in Fig. 26. Only 4 pages of DRAM need to be refreshed.

Corrections are applied into the DRAM.

Data are read towards the EDC checker to test the result of the decoding. The EDC flag is stored into an edc_register[0].

The end of processing is signalled to μP by interrupt. The μP will take care of transferring data to the ATAPI buffer.

The Elab frame in BCA modes is illustrated in Fig. 27.

*Acquisition and Timing Control - BCA heroic-mode*

If sync detection problems are experienced in the acquisition of data, the µP may decide to do the demodulation of BCA by firmware. Setting the HC bit in µP interface, data received from the data acquisition IC are simply stored into the Input RAM. When the last byte is received, the µP begins to read all the blocks from this memory. After carring out demodulation and formatting of the data blocks, data are written back to Input RAM and the decoding process starts (as in normal BCA mode) when the Start bit in the µP interface is set by the µP.

*DVD DESCRAMBLER*

Data are descrambled just before reaching the EDC checker and Output interface. Bytes coming from the embedded DRAM after Reed-Solomon decoding are descrambled "on the fly" and then passed to the EDC checker.

The basic descrambling operation is to forward one byte unchanged if it does not belong to the main data area, or to perform another XOR operation with the right scrambling value to retrieve the original data. The scramble value must be generate internally.

Each already error-corrected and deinterleaved data sector is treated separately (during descrambling), sequentially byte after byte (as part of an entire Ecc_block of data).

The overall architecture of the Descrambler for DVD mode is depicted in Fig. 28.

The main input is the sequential stream of data bytes. The row and column indexes of the current input data byte are needed for algorithm control. From such information principally the Sector_start (first byte of a new data sector) and the Block_start (first byte of a new Ecc block) must be retrieved.

The Desclamber is provided with the 50MHz master clock and with a data_enable signal indicating that an input data has to be processed.

The main output is a stream of descrambled bytes, accompanied by control signals necessary for the EDC checker and output interface.

The Descrambler processes only DVD-data, depending on the format of the input data. In the other modes of operation, the data bytes are simply conveyed to the output.

*CD DESCRAMBLER*

Data (not audio) coming from CIRC decoding must be descrambled before going to C3 decoding. This operation may be done inside ECC-IC or in an ATAPI-IC. A bit written in a register (en_des) enables/disables this function.

The incoming byte are xored with a scramble value generated by a FSR, as shown in the functional block diagram of Fig. 29.

The 15-bit register is of the parallel-block synchronized type and fed back according to the polynomial $x^{15} + x + 1$.

After the Sync of the sector (0x00, 0xFF ten times, 0x00) has been found, the register is preset with the value 0x0001, where 1 is the least significant bit. A symbol counter will count 2340 bytes to be descrambled.

Each time a Sync pattern is detected a preset is generated (the Sync window is always active) and the counter reset to 2340 (Short Sectors).

Each time the counter reaches zero, a preset signal is generated anyway, even if no Sync pattern is found. The counter is reset to 2340 (Long Sector).

Each time a new byte is read, the FSR value St[15:0] is updated as follows:

$S_{t+1}[15] = S_t[8] \text{xor} S_t[7]$

$S_{t+1}[14] = S_t[7] \text{xor} S_t[6]$

$S_{t+1}[13] = S_t[6] \text{xor} S_t[5]$

$S_{t+1}[12] = S_t[5] \text{xor} S_t[4]$

$S_{t+1}[11] = S_t[4] \text{xor} S_t[3]$

$S_{t+1}[10] = S_t[3] \text{xor} S_t[2]$

$S_{t+1}[\ 9] = S_t[2] \text{xor} S_t[1]$

$S_{t+1}[\ 8] = S_t[1] \text{xor} S_t[0]$

$S_{t+1}[7:0] = S_t[15:8]$

*EDC CHECKER*

The EDC is a CRC check which , in DVD-mode, is applied to the sectors after the descrambling has been performed. In BCA mode the EDC check is effected to test the result of decoding.

*BCA Mode*

A decoded BCA block consists of 16xn bytes, with $1 \leq n \leq 12$. The last four bytes are parity bytes, obtained dividing the information bits by the generator polynomial:

$$g(x) = x^{32} + x^{31} + x^4 + 1$$

At the beginning of each block a start signal resets the EDC value to zero. With each enable signal, data is read from the embedded DRAM to the EDC.

The current value of EDC is updated each time a new byte is received.

After 192 bytes the check is stopped and a result flag is set to zero if the check was satisfactory. Otherwise this flag is set to 1. The EDC doesn't change the data in any case.

The result flag of the block is stored into a register edc_reg[0], and an interrupt is generated towards the μP to signal the end of BCA processing.

The μP will take care to read the decoded bytes from internal DRAM to an ATAPI buffer.

*DVD Mode*

Each sector consists of 12 rows*172 bytes. The EDC checker gives a correct check if the error-correction has performed well. Of course, this check has no 100% safety, but a high probability that incorrect data be recognized.

At the beginning of each sector, the start signal resets the EDC value to zero. With each enable signal data is shifted from the DVD Descrambler to the EDC checker.

The current value of EDC is updated each time a new byte is received.

After 2064 bytes the check is stopped and the result flag is set to zero if the check was satisfactory. Otherwise this flag is set to 1.

All the result flags of the sectors inside an Ecc_block of data are stored into a register edc_reg[15:0]; when the first EDC failure occurs, an interrupt is generated towards the μP; the content of this register will be reset when the first byte of the next Ecc_block will be output.

A functional block diagram of the EDC Checker is shown in Fig. 30.

Four bytes are attached to 2060 bytes in a data sector. Let's suppose that the MSB of the first byte of the sector is b16511 and the LSB of the last byte of EDC is b0.

The generator polynomial of the code is

$$g(x) = x^{32} + x^{31} + x^4 + 1$$

and the information polynomial is

$$I(x) = \sum_{i\ =\ 16511}^{32} b_i x^i$$

The implemented algorithm works directly on bytes. This allows to reduce the frequency of the clock needed in this processing.

An internal ROM storing 8 words of 32 bits is used to update the CRC value each time a new byte is received. The stored values are obtained as remainder of the division

$$W_i = R\left(\frac{x^{32+i}}{g(x)}\right)$$

calculated in GF(2).

At the beginning of the sector the value of CRC is initialized to 0x0000. The input byte is xored with the most significant byte of the current CRC value: each "1" in the resulting byte enables one stored value; all the enabled bytes are xored to give an updating value. The new CRC value is obtained shifting the old one and xoring the result with the updating value. This processing is illustrated in the diagram of Fig. 31.

# CLAIMS

1. A method of organizing and transferring data among the functional blocks of an integrated system (ECC-IC) of a read channel of data recorded on DVD-Rom, DVD-Ram and DVD-R for performing Reed-Solomon decoding including off-line heroic correction, said integrated system including an input buffer (INPUT BUFFER), an interface with a microcontroller bus (μP IF), a Reed-Solomon decoder (REED-SOLOMON DECODER), an embedded RAM (DRAM) accessed through a 17-bit bus, a descrambling and EDC control block (DESCRAMBLING & EDC) for DVD modes of operation, a data output interface (OUTPUT INTERFACE) and a timing control block (TIMING CONTROL), characterized in that the decoding of the input data acquired through said input buffer (INPUT BUFFER) is performed at a rate of up to four-times the reference bit rate of DVD formatted data using a clock for accessing said embedded RAM having a frequency half that of the clock that is used in said Reed-Solomon decoder, while reducing the number of accesses to said embedded RAM needed to perform the decoding, the method comprising the steps of:

a) organizing DVD mode input data in two blocks or rows of 182 bytes each;

b) storing each input block in 12 pages of distinct banks in which said embedded RAM (DRAM) is divided, functioning in a synchronous page mode, each bank being split into 512 pages of 32 words of 17 bits, by organizing bytes into words of 16 bits, sequentially readable therefrom without waiting for any precharge cycle;

c) decoding simultaneously two input blocks or rows in said Reed-Solomon decoder (REED-SOLOMON DECODER), correcting errors and storing decoding flags in said embedded RAM (DRAM);

d) refreshing said embedded RAM (DRAM) while the decoding of two blocks is in progress;

e) decoding two columns of 208 bytes each of said stored input blocks by reading said columns word by word from said embedded RAM (DRAM) and performing parallel Reed-Solomon decoding on two columns at the time in said Reed-Solomon decoder (REED-SOLOMON DECODER), correcting errors and storing decoding flags in said embedded RAM (DRAM).

2. A method of organizing and transferring data among the functional blocks of an integrated system (ECC-IC) of a read channel of data recorded on a CD Rom for performing deinterleaving, decoding and correction, said integrated system including an input buffer (INPUT BUFFER), an interface with a microcontroller bus (μP IF), a Reed-Solomon decoder (REED-SOLOMON DECODER), an embedded RAM (DRAM) accessed through a 17-bit bus, a descrambler (DESCRAMBLING CD), a data output interface (OUTPUT INTERFACE) and a timing control block (TIMING CONTROL), characterized in that the processing of the input data acquired through said input buffer (INPUT BUFFER) is performed at a rate of up to 32 times the reference bit rate of CD formatted data using a clock for accessing said embedded RAM having a frequency half that of the clock that is used in said Reed-Solomon decoder, while reducing the number of accesses to said embedded RAM needed to perform the processing, the method comprising the steps of:

a) storing CD mode frames of input data including erasure flags in said input buffer (INPUT BUFFER) and conveying subcode and sync data to said output interface (OUTPUT INTERFACE)

b) performing a first step of deinterleaving (M1-deinterleaving) in said input buffer (INPUT BUFFER) reading frames of input data therefrom, inputting the frames to said Reed-Solomon decoder (REED-SOLOMON DECODER) that

calculates the C1 syndrome and storing the data frames in a correction fifo buffer while correcting errors on the fly of a previously stored frame and storing it in two banks of said embedded RAM;

c) performing a second step of deinterleaving (M2-deinterleaving) in said two banks of three distinct banks in which said embedded RAM is divided, functioning in a synchronous page mode, each bank being split into 512 pages of 32 words of 17 bits, by organizing bytes and C1-decoding flags into words of 17 bits, by storing with a new addressing scheme data in 28 pages subject to refreshing, each page containing two of said frames, by reading frames of deinterleaved data and flags from said two banks, inputting the frames to said Reed-Solomon decoder that calculates the C2 syndrome and storing the data frames in a correction fifo buffer while correcting errors on the fly of a previously stored frame and storing it in a third bank of said embedded RAM;

d) performing a third and last step of deinterleaving (M3-deinterleaving) in the third one of said three distinct banks in which said embedded RAM is divided, using two pages for the process, including the reading four frames of deinterleaved data from said two banks, each byte with its own flag of correct/incorrect decoding, and packing two bytes in one word to be written in the same third bank of said embedded RAM;

e) refreshing 28 pages of each of said three banks of embedded RAM simultaneously while the processing of one frame is in progress;

f) reading frames from said third bank of embedded RAM and inputting them to said descrambler (DESCRAMBLING CD) toward said output interface.

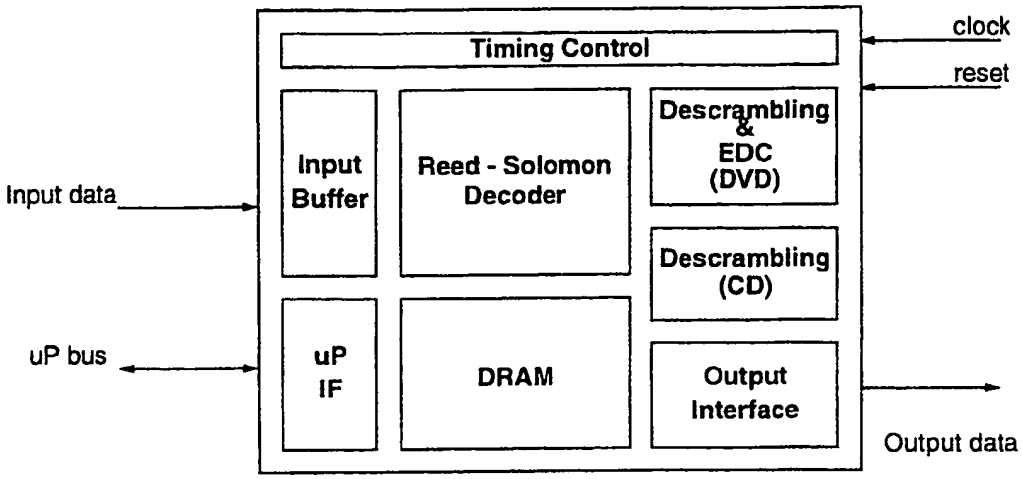

Timing Control
clock
reset
Input data
Input Buffer
Reed - Solomon Decoder
Descrambling & EDC (DVD)
Descrambling (CD)
uP bus
uP IF
DRAM
Output Interface
Output data

FIG.1

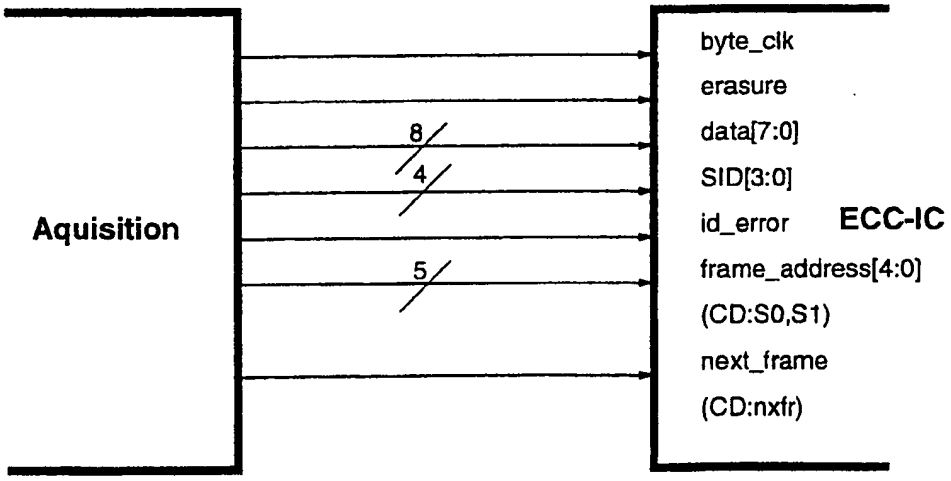

Aquisition
8
4
5
byte_clk
erasure
data[7:0]
SID[3:0]
id_error
ECC-IC
frame_address[4:0]
(CD:S0,S1)
next_frame
(CD:nxfr)

FIG.2

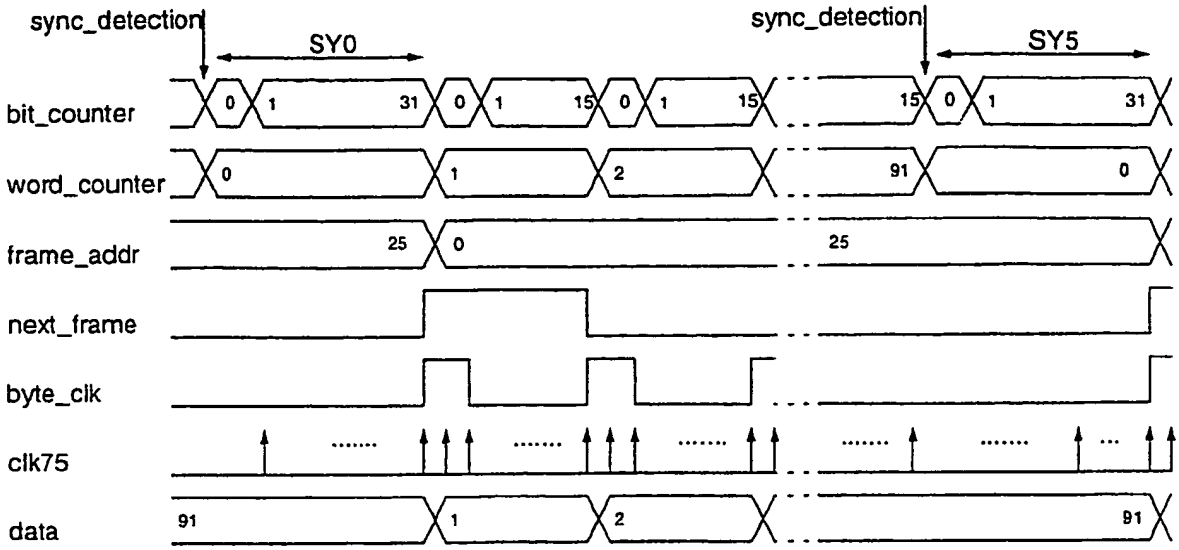

sync_detection
SY0
sync_detection
SY5
bit_counter
word_counter
frame_addr
next_frame
byte_clk
clk75
data

FIG.3

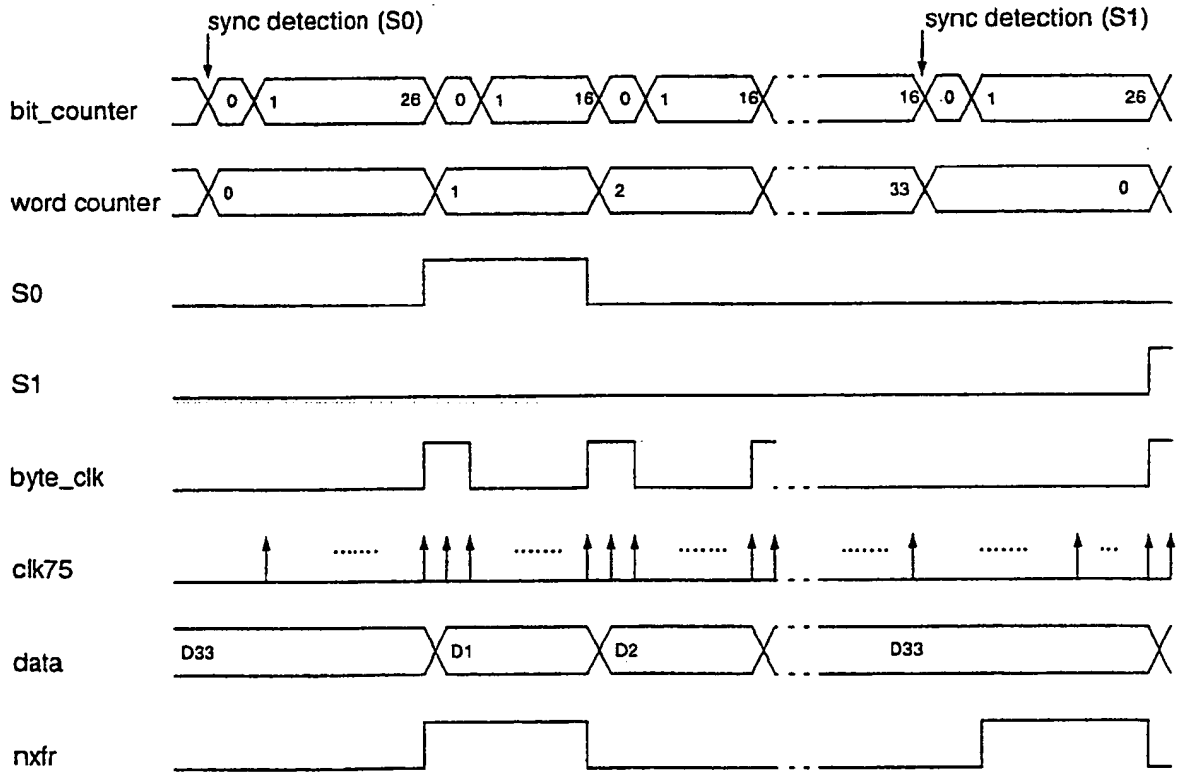

sync detection (S0)
sync detection (S1)
bit_counter
word counter
S0
S1
byte_clk
clk75
data
nxfr

FIG.4

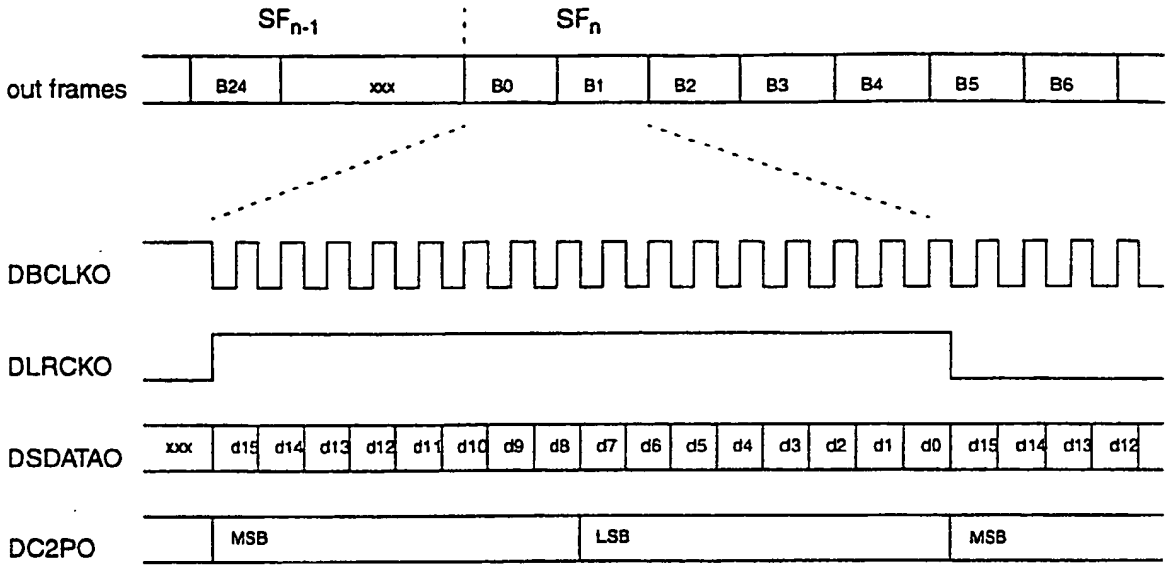

SFn-1
SFn
out frames
B24
xxx
B0
B1
B2
B3
B4
B5
B6
DBCLKO
DLRCKO
DSDATAO
xxx
d15
d14
d13
d12
d11
d10
d9
d8
d7
d6
d5
d4
d3
d2
d1
d0
d15
d14
d13
d12
DC2PO
MSB
LSB
MSB

FIG.5

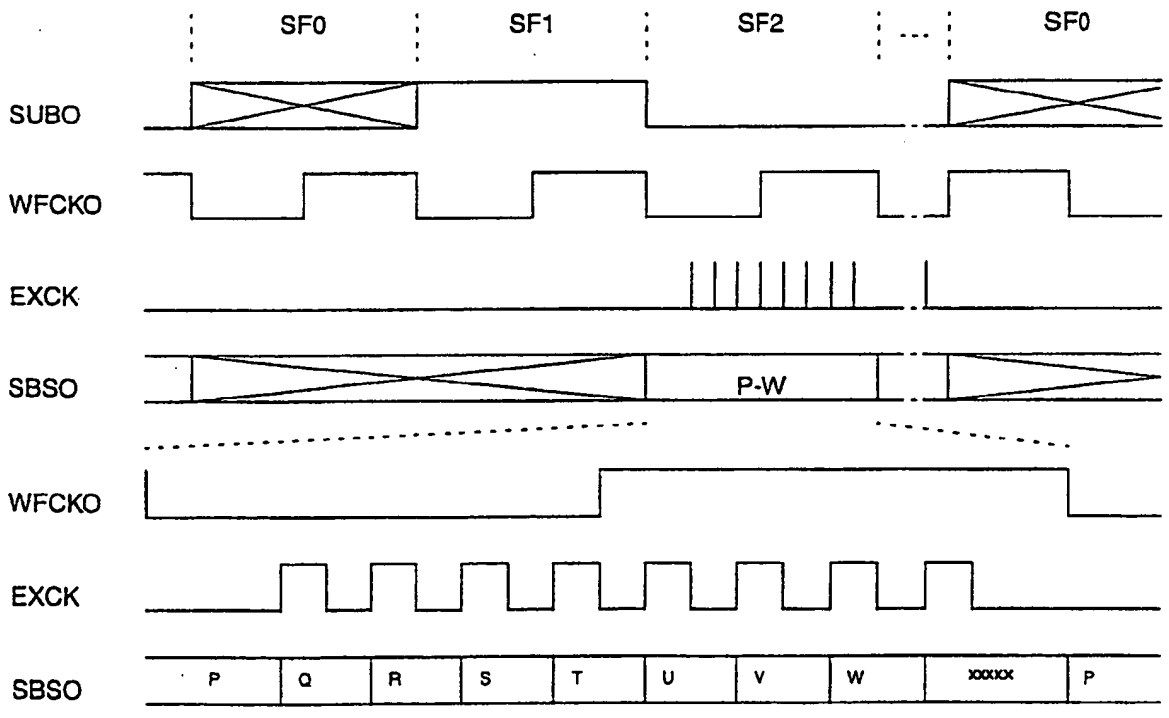

SF0
SF1
SF2
...
SF0
SUBO
WFCKO
EXCK
SBSO
P-W
WFCKO
EXCK
SBSO
P
Q
R
S
T
U
V
W
xxxxx
P

FIG.6

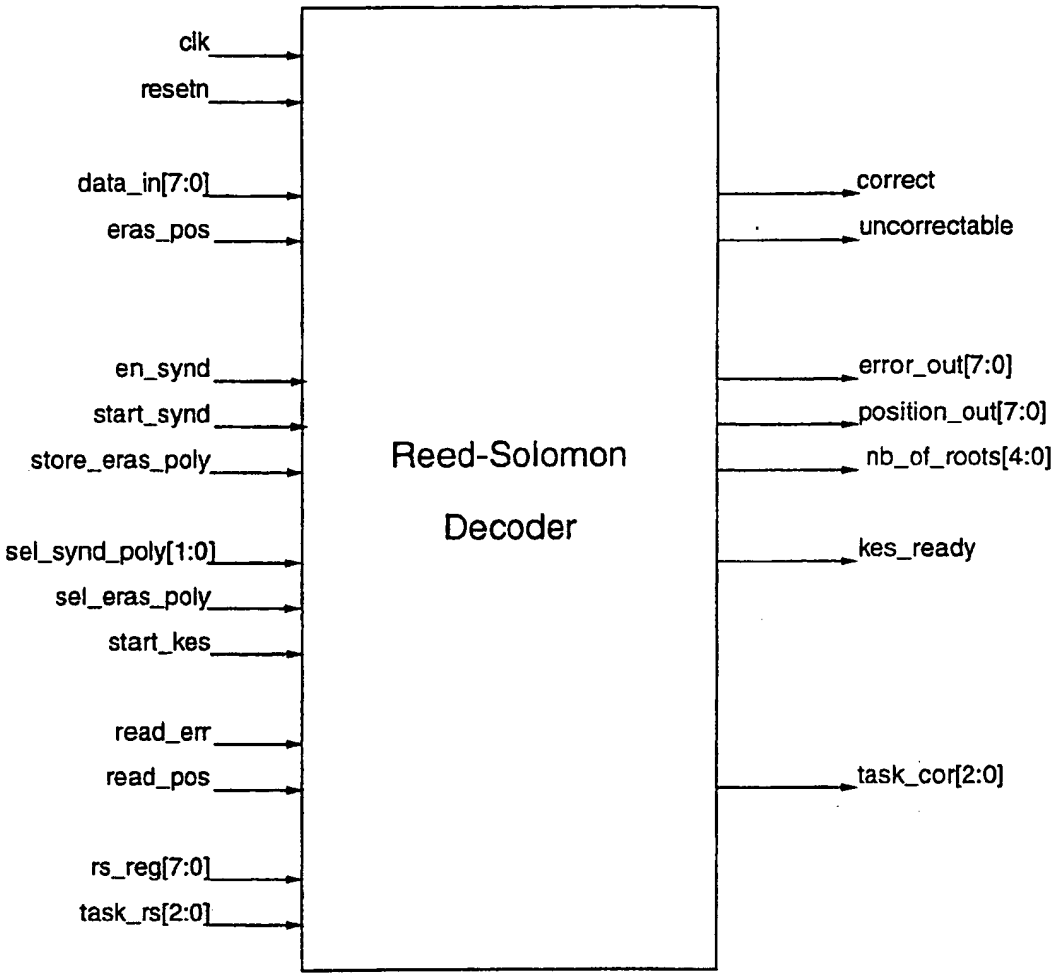

clk
resetn
data_in[7:0]
eras_pos
en_synd
start_synd
store_eras_poly
sel_synd_poly[1:0]
sel_eras_poly
start_kes
read_err
read_pos
rs_reg[7:0]
task_rs[2:0]
Reed-Solomon
Decoder
correct
uncorrectable
error_out[7:0]
position_out[7:0]
nb_of_roots[4:0]
kes_ready
task_cor[2:0]

FIG.7

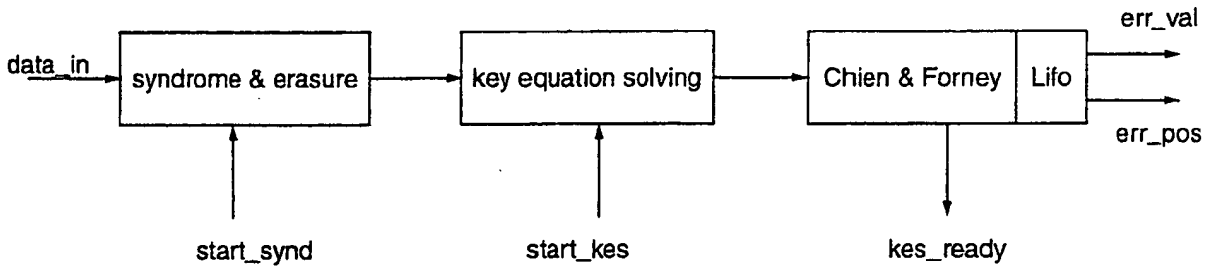

data_in
syndrome & erasure
key equation solving
Chien & Forney
Lifo
err_val
err_pos
start_synd
start_kes
kes_ready

FIG.8

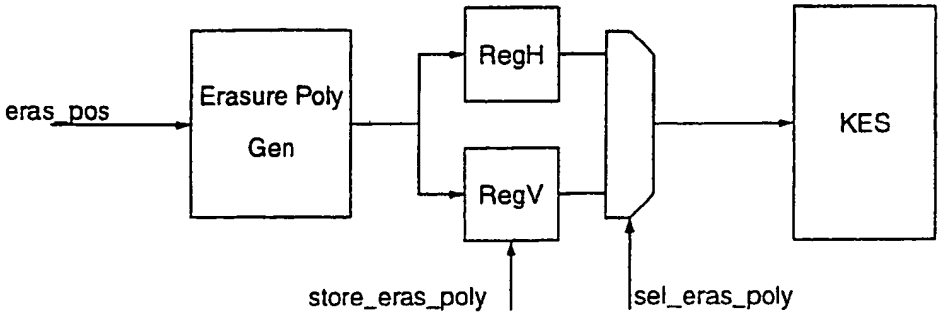

eras_pos
Erasure Poly Gen
RegH
RegV
KES
store_eras_poly
sel_eras_poly

FIG.9

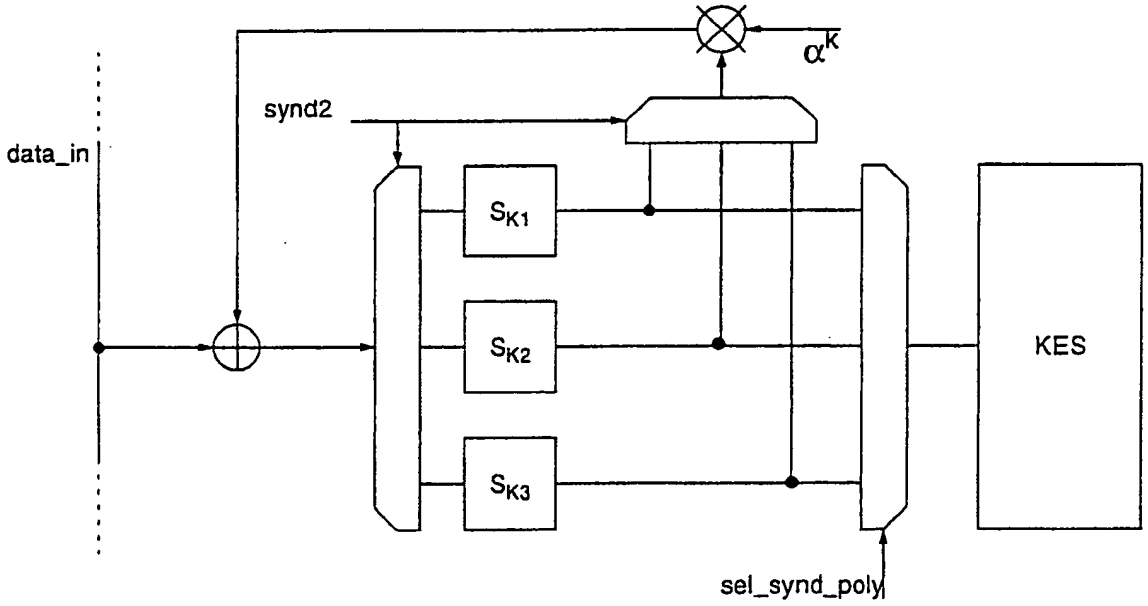

$\alpha^k$
synd2
data_in
$S_{K1}$
$S_{K2}$
$S_{K3}$
KES
sel_synd_poly

FIG.10

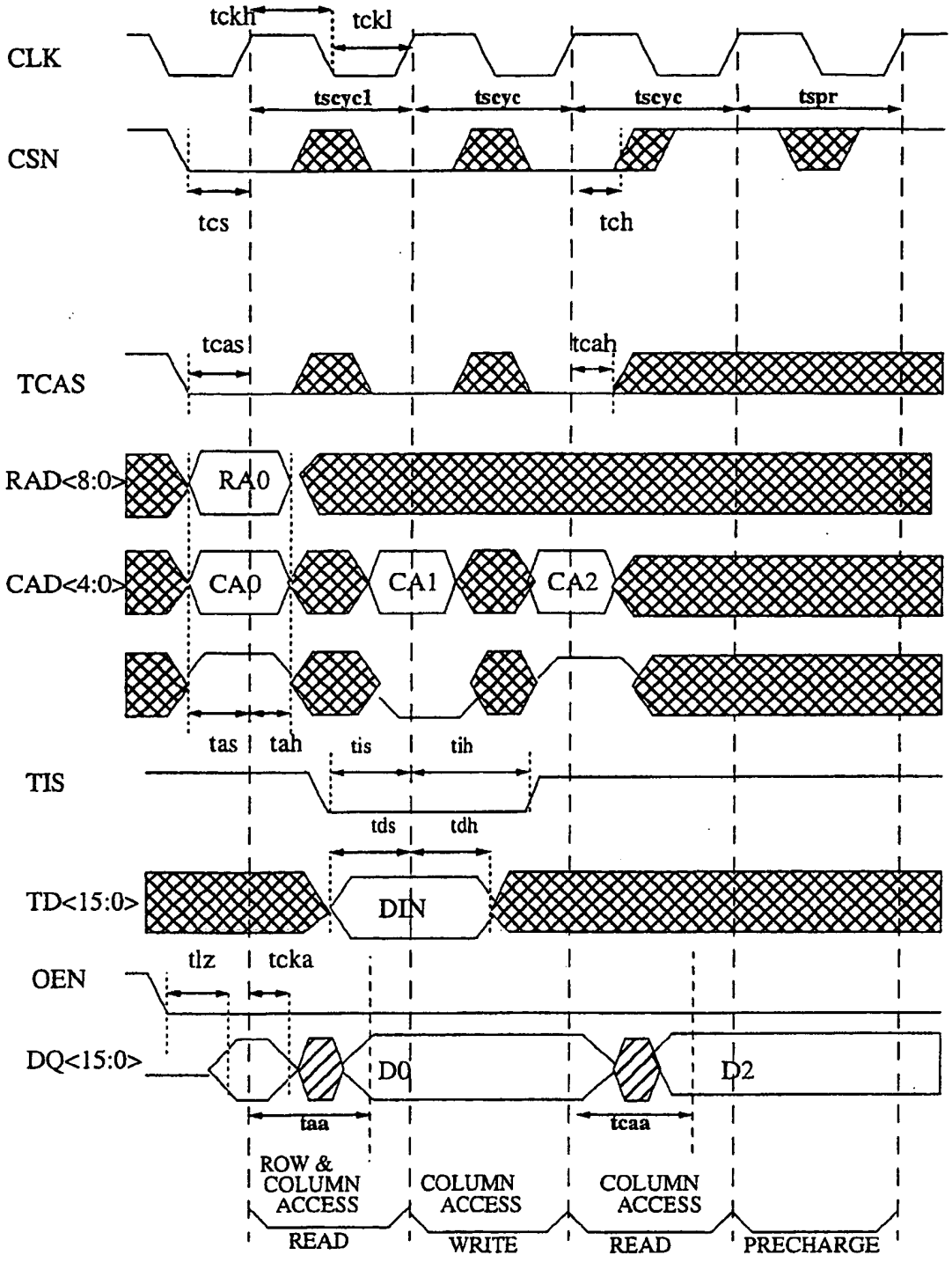

tckh
tckl
CLK
tscyc1
tscyc
tscyc
tspr
CSN
tcs
tch
tcas
tcah
TCAS
RAD<8:0>
RA0
CAD<4:0>
CA0
CA1
CA2
tas
tah
tis
tih
TIS
tds
tdh
TD<15:0>
DIN
tlz
tcka
OEN
DQ<15:0>
D0
D2
taa
tcaa
ROW & COLUMN ACCESS
READ
COLUMN ACCESS
WRITE
COLUMN ACCESS
READ
PRECHARGE

FIG.11

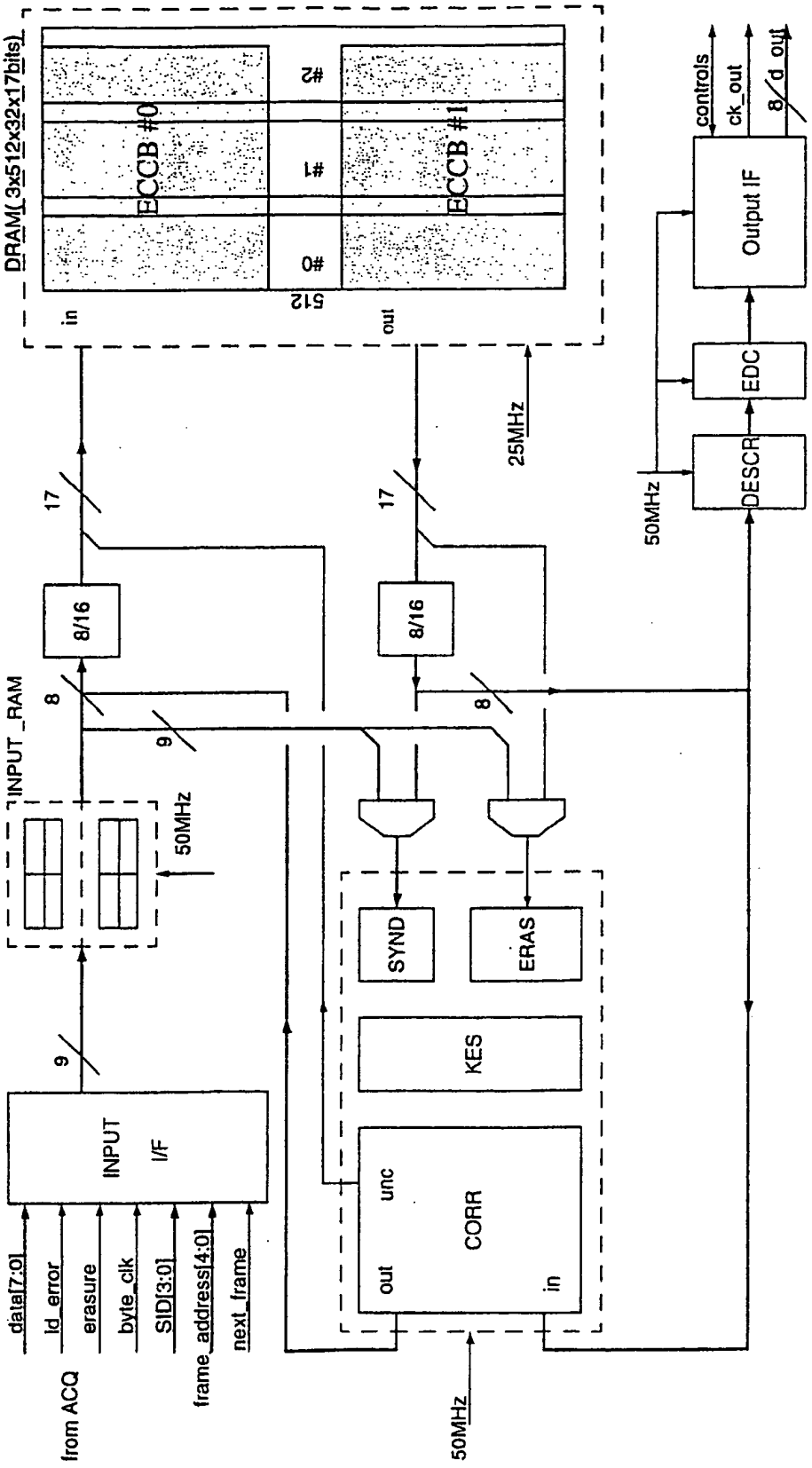

DRAM(3x512x32x17bits)
ECCB #0
ECCB #1
#2
#1
#0
512
in
out
controls
ck_out
8
d_out
Output IF
EDC
DESCR
50MHz
25MHz
17
8/16
8
9
INPUT_RAM
50MHz
SYND
ERAS
KES
CORR
unc
out
in
INPUT I/F
data[7:0]
Id_error
erasure
byte_clk
SID[3:0]
frame_address[4:0]
next_frame
from ACQ
50MHz

FIG.12

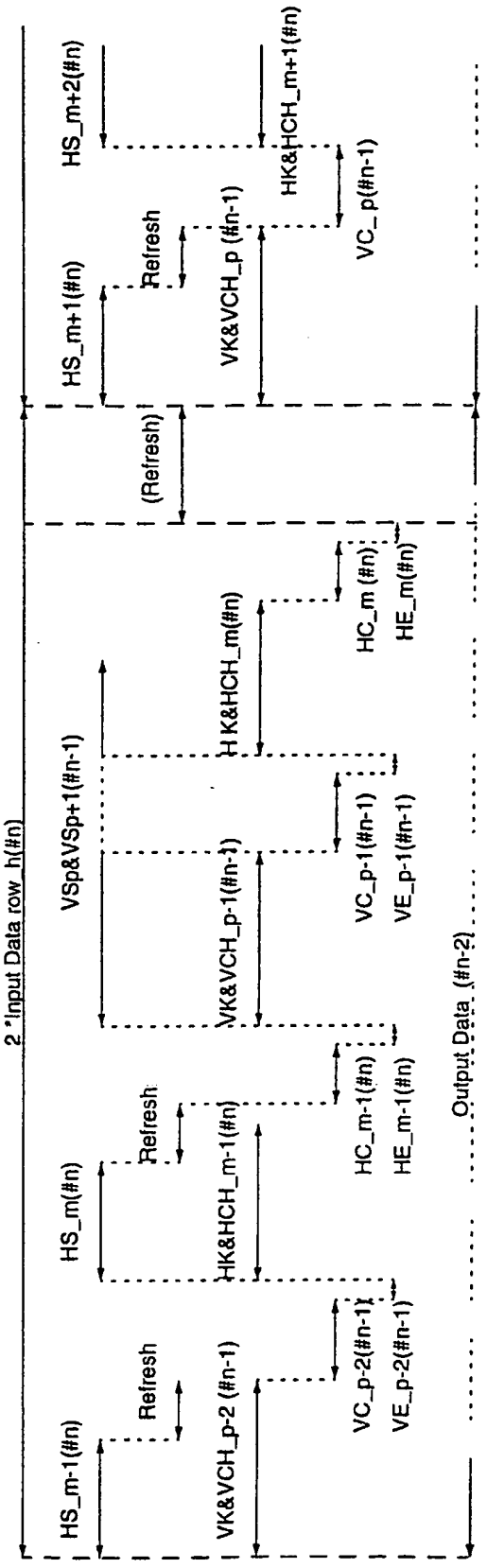

2 *Input Data row_h(#n)
HS_m-1(#n)
Refresh
VK&VCH_p-2 (#n-1)
HS_m(#n)
Refresh
HK&HCH_m-1(#n)
VSp&VSp+1(#n-1)
VK&VCH_p-1(#n-1)
HK&HCH_m(#n)
(Refresh)
HS_m+1(#n)
Refresh
VK&VCH_p (#n-1)
HS_m+2(#n)
HK&HCH_m+1(#n)
VC_p-2(#n-1)
VE_p-2(#n-1)
HC_m-1(#n)
HE_m-1(#n)
VC_p-1(#n-1)
VE_p-1(#n-1)
HC_m (#n)
HE_m(#n)
VC_p(#n-1)
Output Data (#n-2)
HS = Horizontal Syndrome
HK&HCH = Horiz.(Kes+Chien+Forney)
HC = Horizontal Correction
HE = Horizontal Erasure Flag
VS = Vertical Syndrome
VK&VCH = Vert.(Kes+Chien+Forney)
VC = Vertical Correction
VE = Vertical Erasure Flag

FIG.13

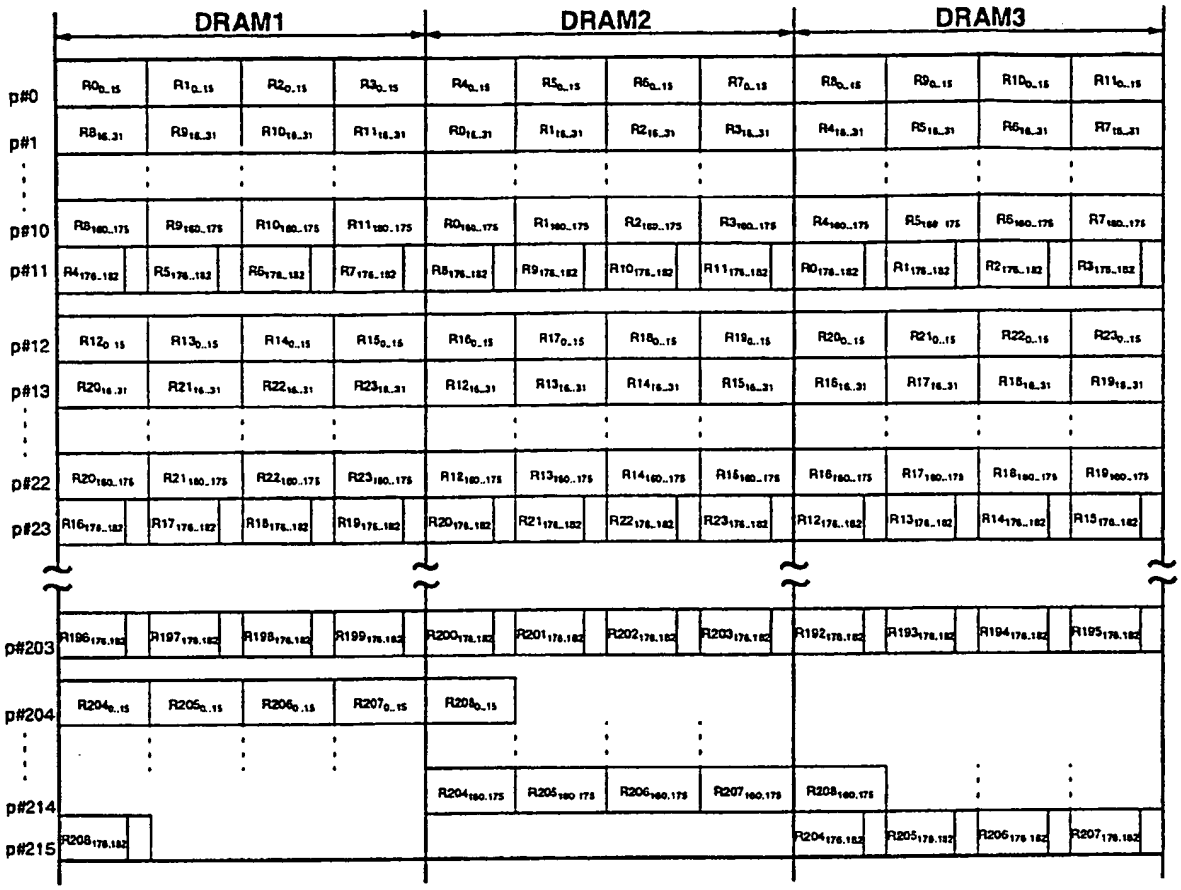

DRAM1
DRAM2
DRAM3
p#0
p#1
p#10
p#11
p#12
p#13
p#22
p#23
p#203
p#204
p#214
p#215

FIG.14

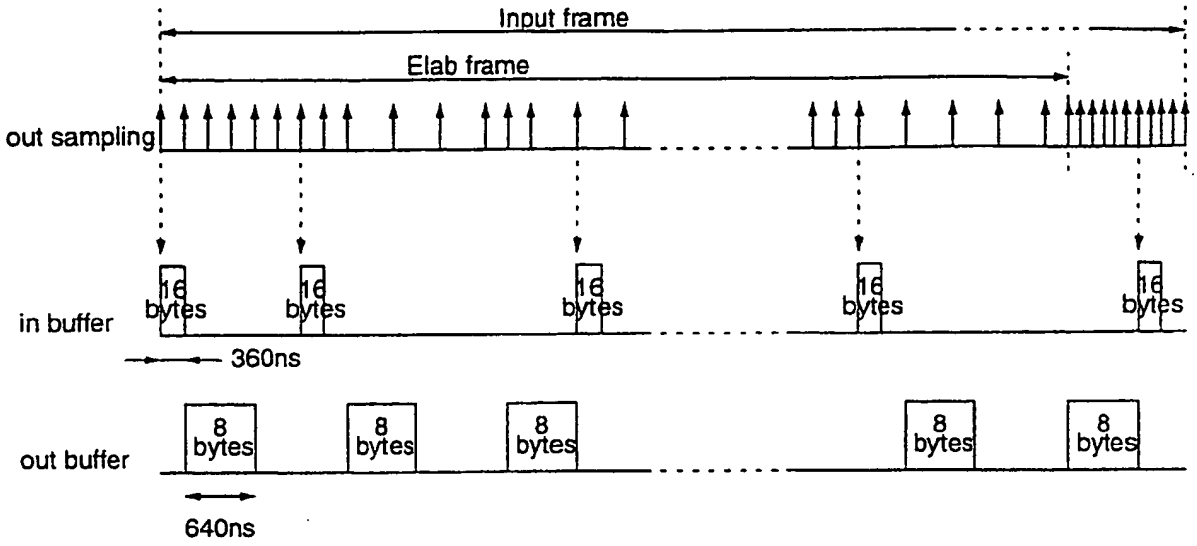

Input frame
Elab frame
out sampling
in buffer
16 bytes
16 bytes
16 bytes
16 bytes
16 bytes
360ns
out buffer
8 bytes
8 bytes
8 bytes
8 bytes
8 bytes
640ns

FIG.15

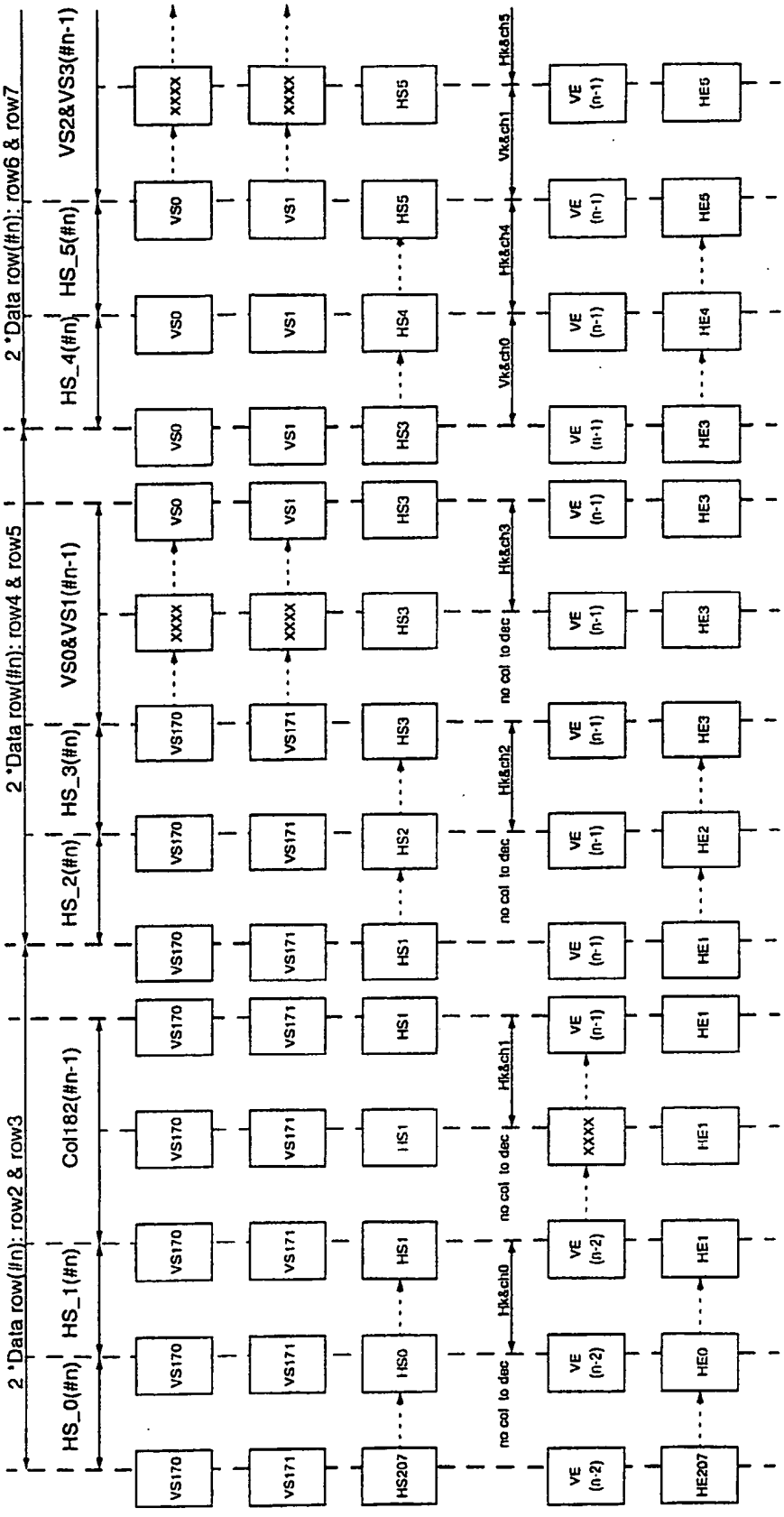

2 *Data row(#n): row2 & row3
2 *Data row(#n): row4 & row5
2 *Data row(#n): row6 & row7
HS_0(#n)
HS_1(#n)
Col182(#n-1)
HS_2(#n)
HS_3(#n)
VS0&VS1(#n-1)
HS_4(#n)
HS_5(#n)
VS2&VS3(#n-1)
VS170
VS171
HS207
HS0
HS1
HS2
HS3
HE207
HE0
HE1
HE2
HE3
HE4
HE5
VE (n-2)
VE (n-1)
XXXX
VS0
VS1
Hk&ch0
Hk&ch1
Hk&ch2
Hk&ch3
Hk&ch4
Hk&ch5
no col to dec

FIG.16

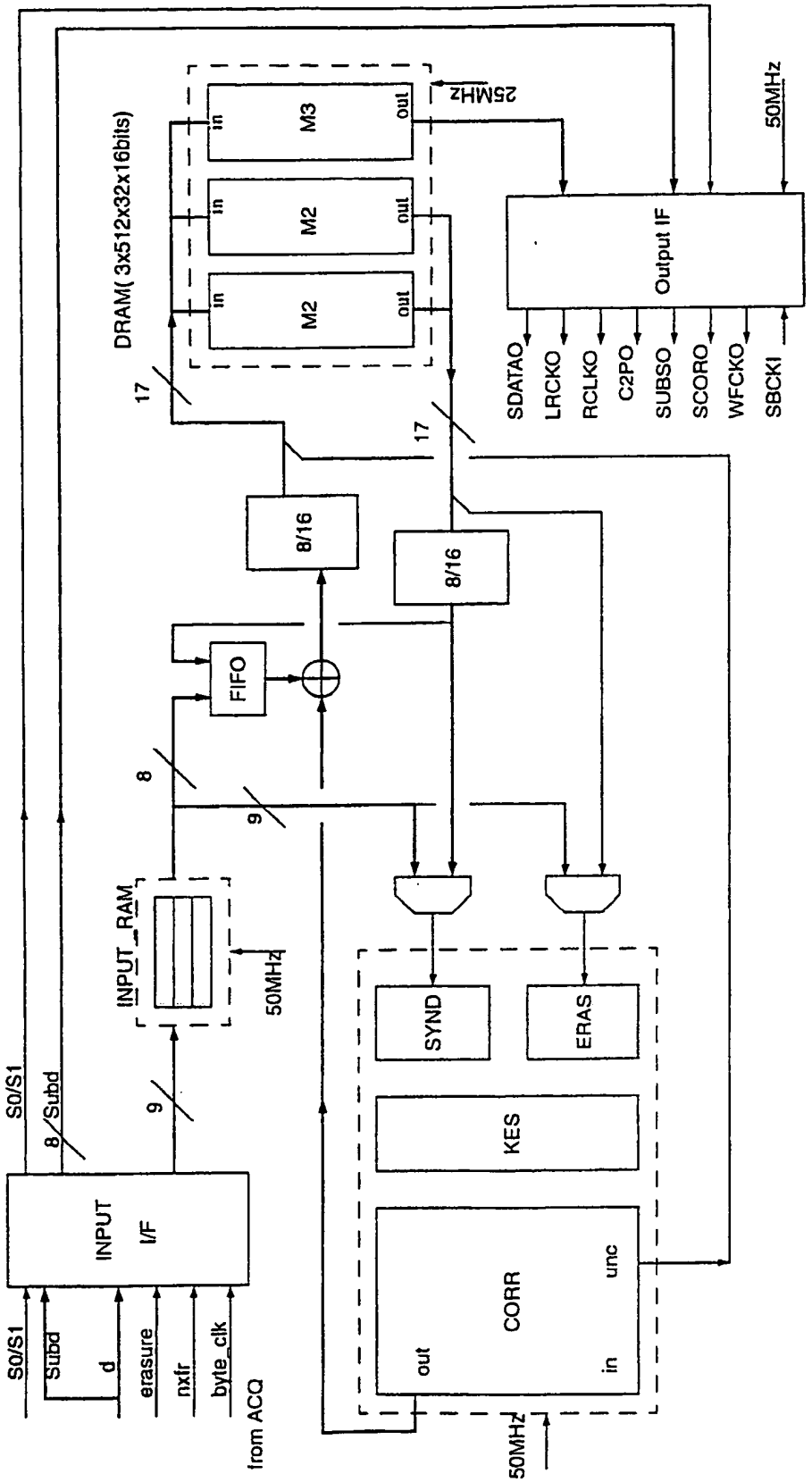
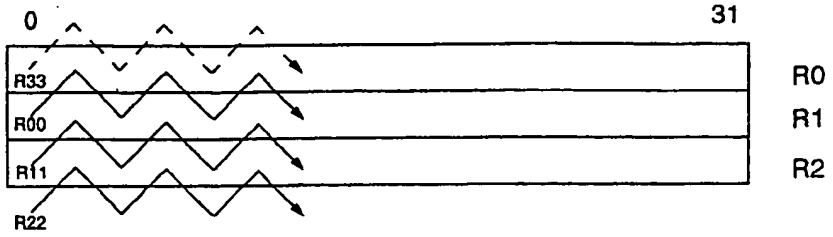

S0/S1
Subd
d
erasure
nxlr
byte_clk
from ACQ
INPUT I/F
8
S0/S1
Subd
9
INPUT_RAM
50MHz
8
9
FIFO
8/16
17
DRAM( 3x512x32x16bits)
M2
M2
M3
in
out
25MHz
50MHz
Output IF
SDATAO
LRCKO
RCLKO
C2PO
SUBSO
SCORO
WFCKO
SBCKI
8/16
17
SYND
ERAS
KES
CORR
out
in
unc
50MHz

FIG.17




write R2 read R00 = $R1_0\ R0_1\ R1_2\ R0_3$ ..... $R1_{28}\ R0_{29}\ R1_{30}\ R0_{31}$

write R0 read R11 = $R2_0\ R1_1\ R2_2\ R1_3$ ..... $R2_{28}\ R1_{29}\ R2_{30}\ R1_{31}$

write R1 read R22 = $R0_0\ R2_1\ R0_2\ R2_3$ ..... $R0_{28}\ R2_{29}\ R0_{30}\ R2_{31}$

write R2 read R33 = $R1_0\ R0_1\ R1_2\ R0_3$ ..... $R1_{28}\ R0_{29}\ R1_{30}\ R0_{31}$

FIG.18

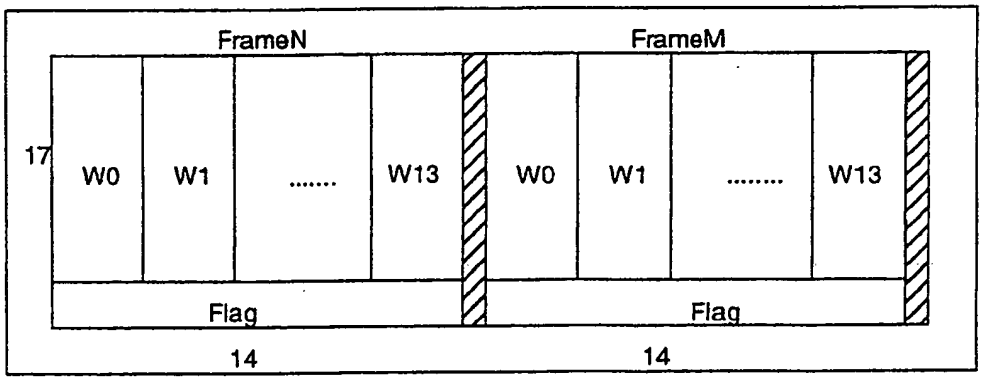



FIG.19

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| F0 | [1 | 5 | 9 | 13 | 17 | 21 | 25 | 29 | 33 | 37 | 41 | 45 | 49 | 53 | 57 | 61 | 65 | 69 | 73 | 77 | 81 | 85 | 89 | 93 | 97 | 101 | 105 | 109] | 1 | 5 | 9 | 13 |
| F1 | [2 | 6 | 10 | 14 | 18 | 22 | 26 | 30 | 34 | 38 | 42 | 46 | 50 | 54 | 58 | 62 | 66 | 70 | 74 | 78 | 82 | 86 | 90 | 94 | 98 | 102 | 106 | 110] | 2 | 6 | 10 | 14 |
| F2 | [3 | 7 | 11 | 15 | 19 | 23 | 27 | 31 | 35 | 39 | 43 | 47 | 51 | 55 | 59 | 63 | 67 | 71 | 75 | 79 | 83 | 87 | 91 | 95 | 99 | 103 | 107 | 111] | 3 | 7 | 11 | 15 |
| F3 | [4 | 8 | 12 | 16 | 20 | 24 | 28 | 32 | 36 | 40 | 44 | 48 | 52 | 56 | 60 | 64 | 68 | 72 | 76 | 80 | 84 | 88 | 92 | 96 | 100 | 104 | 108 | 112] | 4 | 8 | 12 | 16 |
| F4 | 113 | [5 | 9 | 13 | 17 | 21 | 25 | 29 | 33 | 37 | 41 | 45 | 49 | 53 | 57 | 61 | 65 | 69 | 73 | 77 | 81 | 85 | 89 | 93 | 97 | 101 | 105 | 109 | 113] | 5 | 9 | 13 |
| F5 | 114 | [6 | 10 | 14 | 18 | 22 | 26 | 30 | 34 | 38 | 42 | 46 | 50 | 54 | 58 | 62 | 66 | 70 | 74 | 78 | 82 | 86 | 90 | 94 | 98 | 102 | 106 | 110 | 114] | 6 | 10 | 14 |
| F6 | 115 | [7 | 11 | 15 | 19 | 23 | 27 | 31 | 35 | 39 | 43 | 47 | 51 | 55 | 59 | 63 | 67 | 71 | 75 | 79 | 83 | 87 | 91 | 95 | 99 | 103 | 107 | 111 | 115] | 7 | 11 | 15 |
| F7 | 116 | [8 | 12 | 16 | 20 | 24 | 28 | 32 | 36 | 40 | 44 | 48 | 52 | 56 | 60 | 64 | 68 | 72 | 76 | 80 | 84 | 88 | 92 | 96 | 100 | 104 | 108 | 112 | 116] | 8 | 12 | 16 |
| F8 | 113 | 117 | [9 | 13 | 17 | 21 | 25 | 29 | 33 | 37 | 41 | 45 | 49 | 53 | 57 | 61 | 65 | 69 | 73 | 77 | 81 | 85 | 89 | 93 | 97 | 101 | 105 | 109 | 113 | 117] | 9 | 13 |
| F9 | 114 | 118 | [10 | 14 | 18 | 22 | 26 | 30 | 34 | 38 | 42 | 46 | 50 | 54 | 58 | 62 | 66 | 70 | 74 | 78 | 82 | 86 | 90 | 94 | 98 | 102 | 106 | 110 | 114 | 118] | 10 | 14 |
| F10 | 115 | 119 | [11 | 15 | 19 | 23 | 27 | 31 | 35 | 39 | 43 | 47 | 51 | 55 | 59 | 63 | 67 | 71 | 75 | 79 | 83 | 87 | 91 | 95 | 99 | 103 | 107 | 111 | 115 | 119] | 11 | 15 |
| F11 | 116 | 120 | [12 | 16 | 20 | 24 | 28 | 32 | 36 | 40 | 44 | 48 | 52 | 56 | 60 | 64 | 68 | 72 | 76 | 80 | 84 | 88 | 92 | 96 | 100 | 104 | 108 | 112 | 116 | 120] | 12 | 16 |
| F12 | 113 | 117 | 121 | [13 | 17 | 21 | 25 | 29 | 33 | 37 | 41 | 45 | 49 | 53 | 57 | 61 | 65 | 69 | 73 | 77 | 81 | 85 | 89 | 93 | 97 | 101 | 105 | 109 | 113 | 117 | 121] | 13 |
| F13 | 114 | 118 | 122 | [14 | 18 | 22 | 26 | 30 | 34 | 38 | 42 | 46 | 50 | 54 | 58 | 62 | 66 | 70 | 74 | 78 | 82 | 86 | 90 | 94 | 98 | 102 | 106 | 110 | 114 | 118 | 122] | 14 |
| F14 | 115 | 119 | 123 | [15 | 19 | 23 | 27 | 31 | 35 | 39 | 43 | 47 | 51 | 55 | 59 | 63 | 67 | 71 | 75 | 79 | 83 | 87 | 91 | 95 | 99 | 103 | 107 | 111 | 115 | 119 | 123] | 15 |
| F15 | 116 | 120 | 124 | [16 | 20 | 24 | 28 | 32 | 36 | 40 | 44 | 48 | 52 | 56 | 60 | 64 | 68 | 72 | 76 | 80 | 84 | 88 | 92 | 96 | 100 | 104 | 108 | 112 | 116 | 120 | 124] | 16 |
| F16 | 113 | 117 | 121 | 125 | [17 | 21 | 25 | 29 | 33 | 37 | 41 | 45 | 49 | 53 | 57 | 61 | 65 | 69 | 73 | 77 | 81 | 85 | 89 | 93 | 97 | 101 | 105 | 109 | 113 | 117 | 121 | 125] |
| F17 | 114 | 118 | 122 | 126 | [18 | 22 | 26 | 30 | 34 | 38 | 42 | 46 | 50 | 54 | 58 | 62 | 66 | 70 | 74 | 78 | 82 | 86 | 90 | 94 | 98 | 102 | 106 | 110 | 114 | 118 | 122 | 126] |
| F18 | 115 | 119 | 123 | 127 | [19 | 23 | 27 | 31 | 35 | 39 | 43 | 47 | 51 | 55 | 59 | 63 | 67 | 71 | 75 | 79 | 83 | 87 | 91 | 95 | 99 | 103 | 107 | 111 | 115 | 119 | 123 | 127] |
| F19 | 116 | 120 | 124 | 128 | [20 | 24 | 28 | 32 | 36 | 40 | 44 | 48 | 52 | 56 | 60 | 64 | 68 | 72 | 76 | 80 | 84 | 88 | 92 | 96 | 100 | 104 | 108 | 112 | 116 | 120 | 124 | 128] |

FIG.20

Table 1:

| page | DRAM1 | | DRAM2 | |
|---|---|---|---|---|
| 0 | 1 | 9 | 5 | 13 |
| 1 | 2 | 10 | 6 | 14 |
| 2 | 3 | 11 | 7 | 15 |
| 3 | 4 | 12 | 8 | 16 |
| 4 | 17 | 25 | 21 | 29 |
| 5 | 18 | 26 | 22 | 30 |
| 6 | 19 | 27 | 23 | 31 |
| 7 | 20 | 28 | 24 | 32 |
| 8 | 33 | 41 | 37 | 45 |
| 9 | 34 | 42 | 38 | 46 |
| 10 | 35 | 43 | 39 | 47 |
| 11 | 36 | 44 | 40 | 48 |
| 12 | 49 | 57 | 53 | 61 |
| 13 | 50 | 58 | 54 | 62 |
| 14 | 51 | 59 | 55 | 63 |
| 15 | 52 | 60 | 56 | 64 |
| 16 | 65 | 73 | 69 | 77 |
| 17 | 66 | 74 | 70 | 78 |
| 18 | 67 | 75 | 71 | 79 |
| 19 | 68 | 76 | 72 | 80 |
| 20 | 81 | 89 | 85 | 93 |
| 21 | 82 | 90 | 86 | 94 |
| 22 | 83 | 91 | 87 | 95 |
| 23 | 84 | 92 | 88 | 96 |
| 24 | 97 | 105 | 101 | 109 |
| 25 | 98 | 106 | 102 | 110 |
| 26 | 99 | 107 | 103 | 111 |
| 27 | 100 | 108 | 104 | 112 |

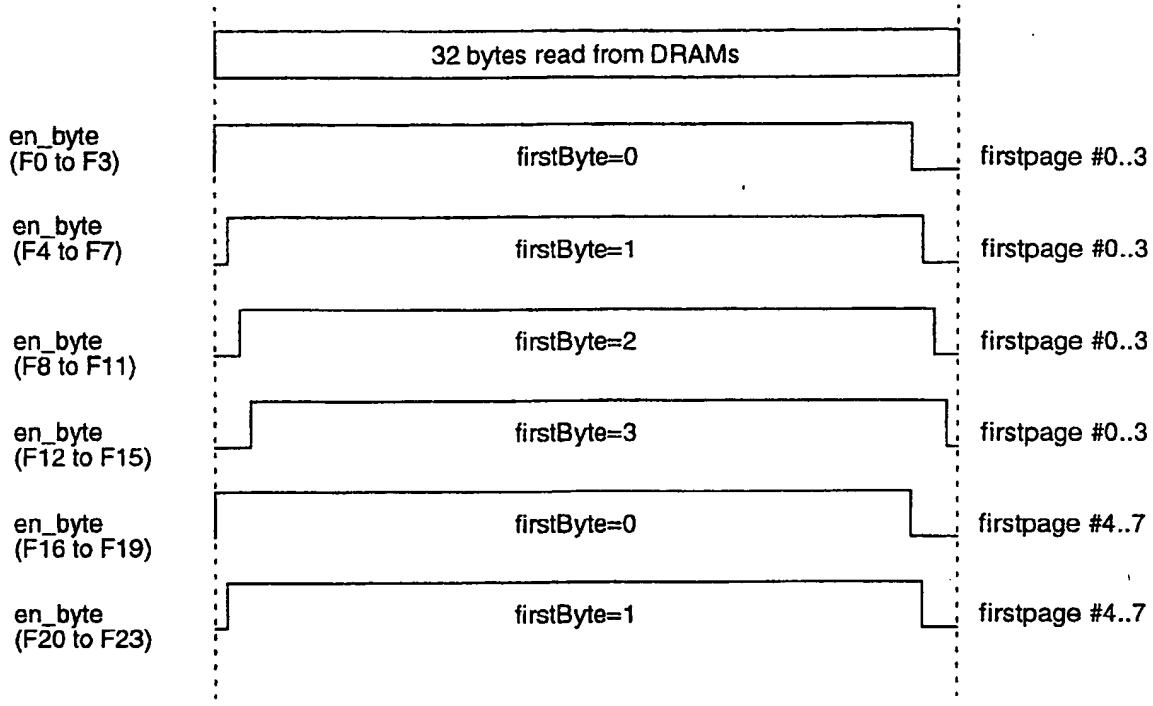

32 bytes read from DRAMs
en_byte (F0 to F3)
firstByte=0
firstpage #0..3
en_byte (F4 to F7)
firstByte=1
firstpage #0..3
en_byte (F8 to F11)
firstByte=2
firstpage #0..3
en_byte (F12 to F15)
firstByte=3
firstpage #0..3
en_byte (F16 to F19)
firstByte=0
firstpage #4..7
en_byte (F20 to F23)
firstByte=1
firstpage #4..7

FIG.21

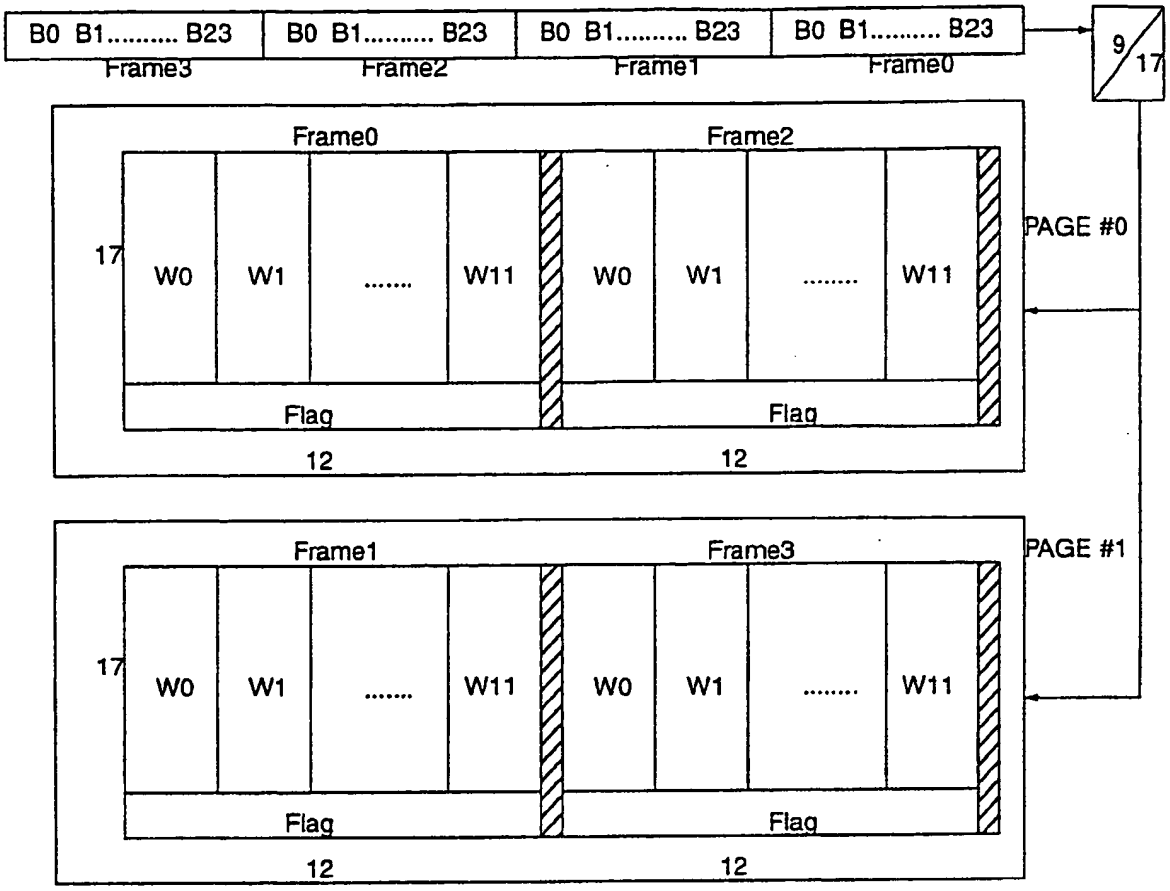
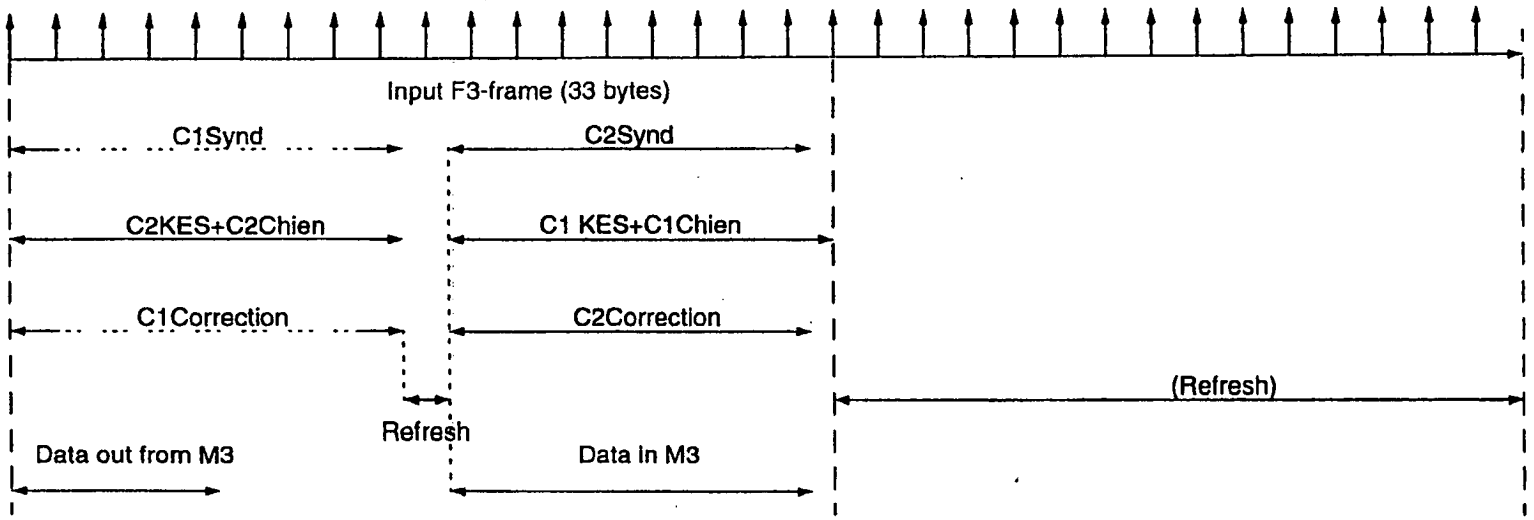

B0 B1..... B23
B0 B1..... B23
B0 B1..... B23
B0 B1..... B23
Frame3
Frame2
Frame1
Frame0
9/17
Frame0
Frame2
17
W0
W1
.....
W11
W0
W1
.....
W11
PAGE #0
Flag
Flag
12
12
Frame1
Frame3
17
W0
W1
.....
W11
W0
W1
.....
W11
PAGE #1
Flag
Flag
12
12

FIG.22




C1 Syndrome: Read from Input RAM, store into Correction Fifo, read from Correction Fifo, store into DRAM1 or DRAM2

(when Input RAM requires to store the incoming data the syndrome updating is disabled)

C2 Syndrome: Read from DRAM1 and DRAM2, store into Correction Fifo, read from Correction FIFO, store into DRAM3

Data out from M3: Read from DRAM3. This process can be parallel with C1 Syndrome update.

Data in M3: a byte is coming from correction Fifo as soon as a byte is read from M2 memory

**FIG.23**

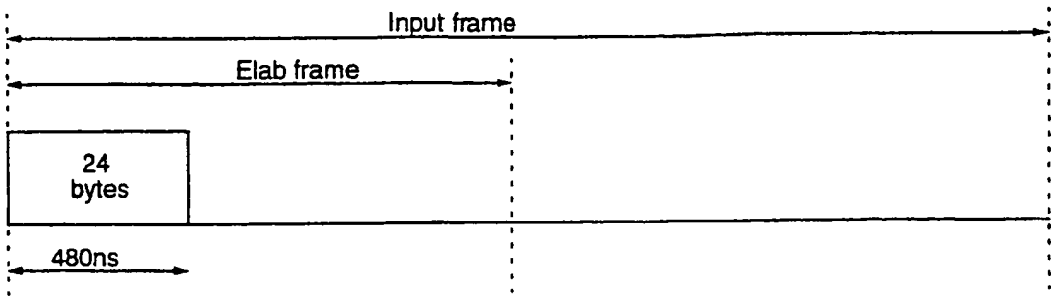

Input frame
Elab frame
24
bytes
480ns

FIG.24

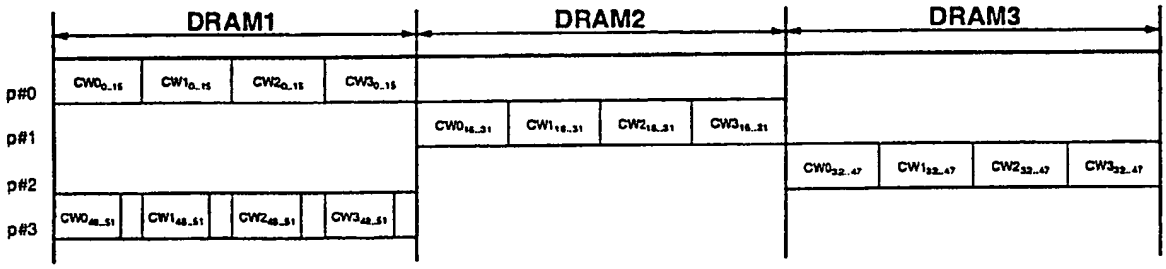

DRAM1
DRAM2
DRAM3
p#0
p#1
p#2
p#3

FIG.26

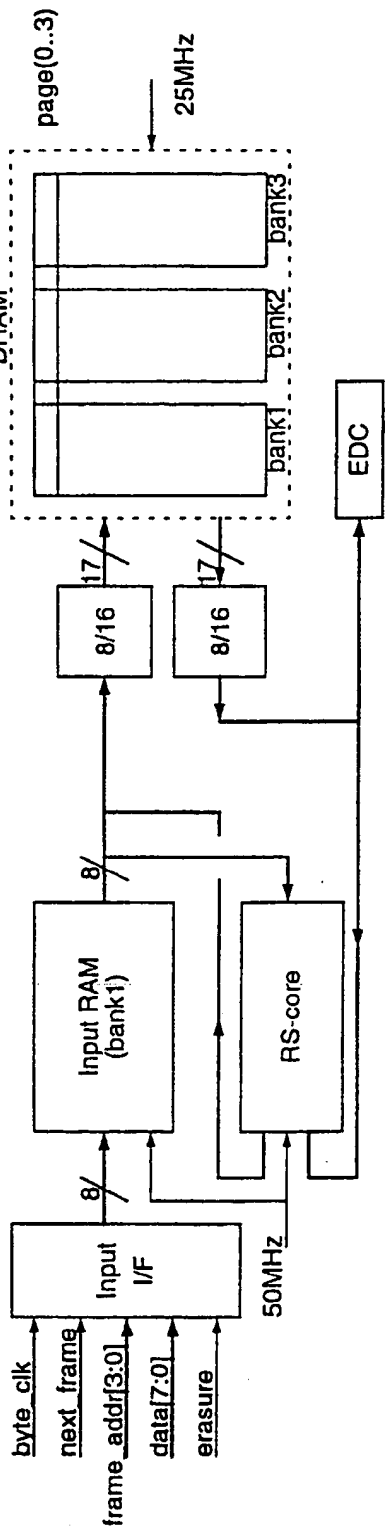

page(0..3)
25MHz
DRAM
bank3
bank2
bank1
EDC
17
17
8/16
8/16
8/
Input RAM (bank1)
RS-core
8/
Input I/F
50MHz
byte_clk
next_frame
frame_addr[3:0]
data[7:0]
erasure

FIG.25

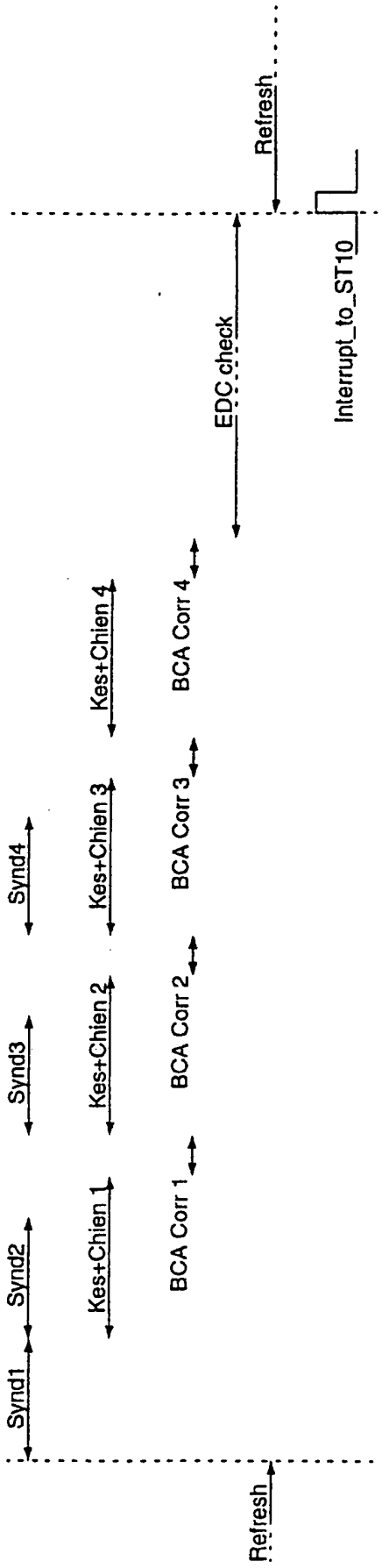

Refresh
EDC check
Interrupt_to_ST10
Kes+Chien 4
Kes+Chien 3
Kes+Chien 2
Kes+Chien 1
BCA Corr 4
BCA Corr 3
BCA Corr 2
BCA Corr 1
Synd4
Synd3
Synd2
Synd1
Refresh

FIG.27

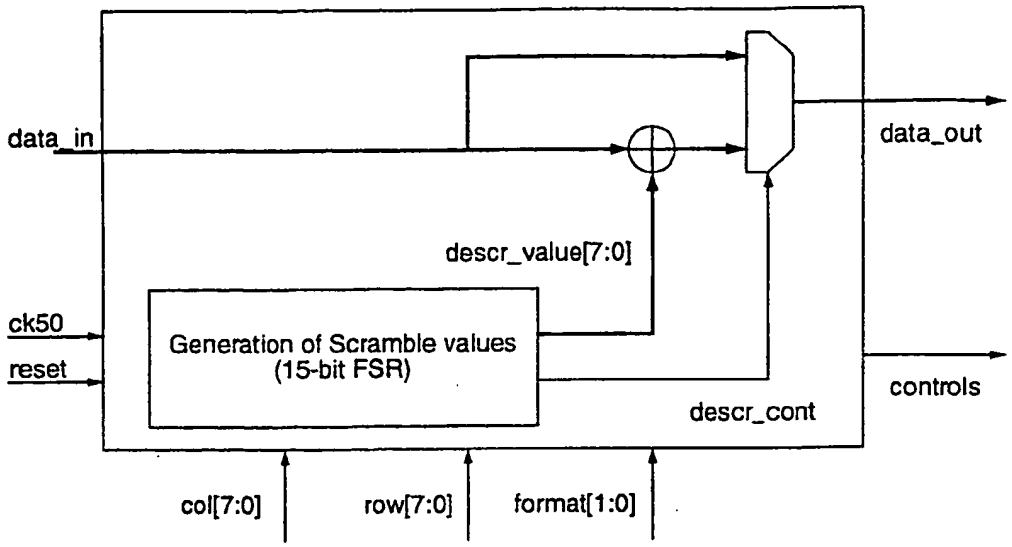

data_in
data_out
descr_value[7:0]
ck50
reset
Generation of Scramble values
(15-bit FSR)
controls
descr_cont
col[7:0]
row[7:0]
format[1:0]

FIG.28

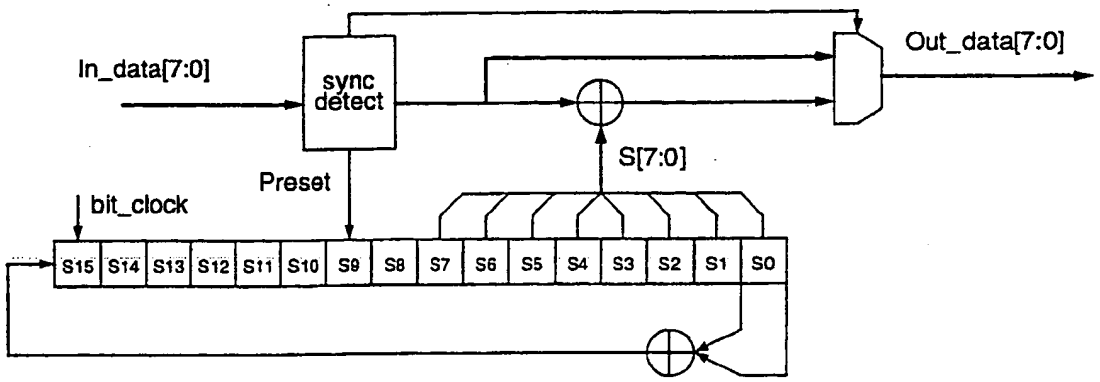

In_data[7:0]
sync
detect
Out_data[7:0]
S[7:0]
Preset
bit_clock
S15
S14
S13
S12
S11
S10
S9
S8
S7
S6
S5
S4
S3
S2
S1
S0

FIG.29

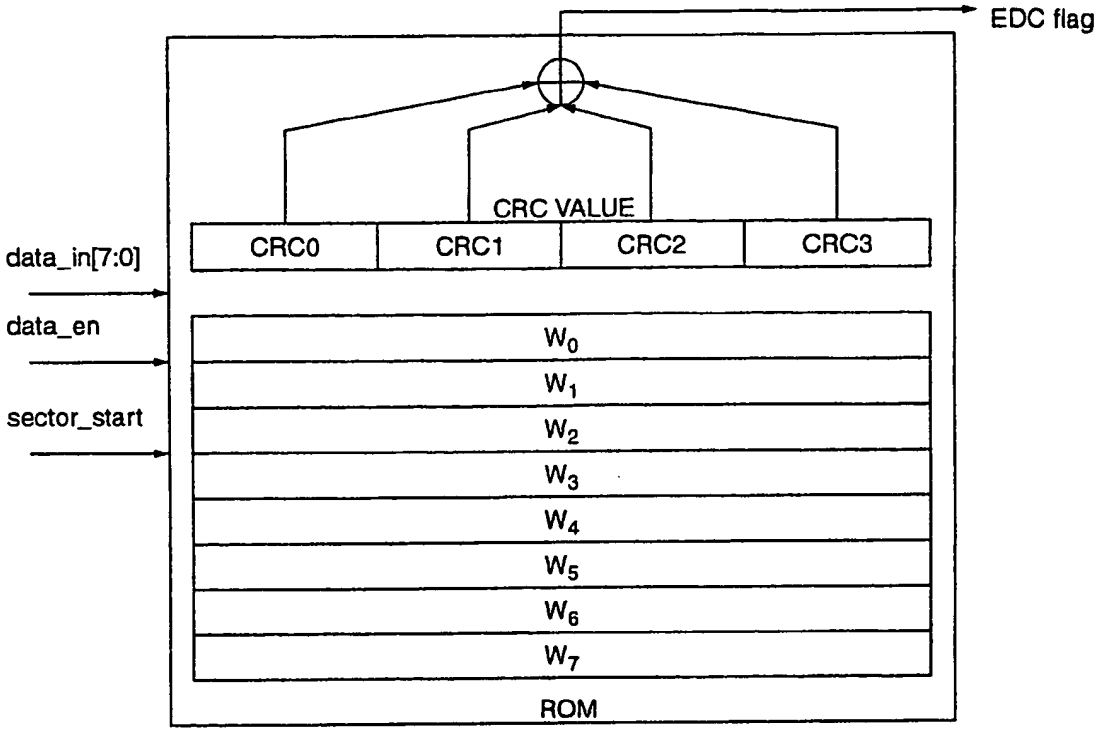

EDC flag
CRC VALUE
CRC0
CRC1
CRC2
CRC3
data_in[7:0]
data_en
sector_start
W0
W1
W2
W3
W4
W5
W6
W7
ROM

FIG.30

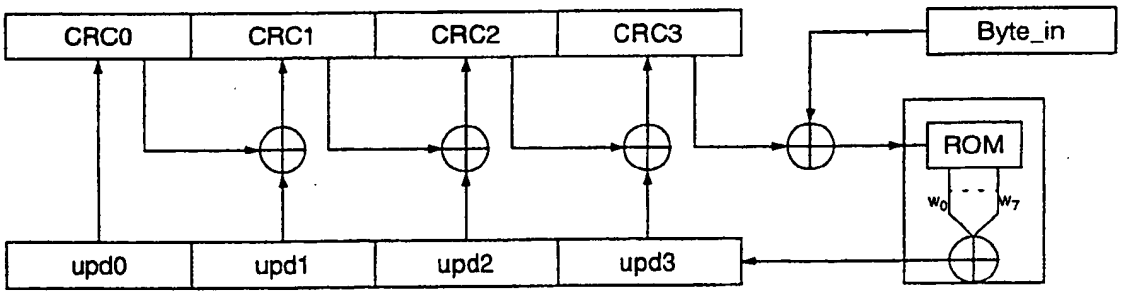

CRC0
CRC1
CRC2
CRC3
Byte_in
ROM
W0
W7
upd0
upd1
upd2
upd3

FIG.31

# INTERNATIONAL SEARCH REPORT

Inte onal Application No
PCT/IT 98/00056

A. CLASSIFICATION OF SUBJECT MATTER
IPC 6 G11B20/18

According to International Patent Classification(IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC 6 G11B H03M

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

| Category ° | Citation of document, with indication, where appropriate, of the relevant passages | Relevant to claim No. |
|---|---|---|
| A | EP 0 821 493 A (SGS THOMSON MICROELECTRONICS) 28 January 1998<br>see page 7, line 2 - page 8, line 4; figures 4,5<br>see page 9, line 2 - page 10, line 25; figure 7 | 1 |
| A | US 5 691 994 A (ACOSTA MARC ET AL) 25 November 1997<br>see column 1, line 36 - column 2, line 52 | 1 |
| A | US 5 631 914 A (KASHIDA MOTOKAZU ET AL) 20 May 1997<br>see column 5A, line 50 - column 6, line 46; figure 2 | 1 |

-/--

[X] Further documents are listed in the continuation of box C.

[X] Patent family members are listed in annex.

° Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publicationdate of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

| Date of the actual completion of theinternational search | Date of mailing of the international search report |
|---|---|
| 24 November 1998 | 01/12/1998 |

| Name and mailing address of the ISA | Authorized officer |
|---|---|
| European Patent Office, P.B. 5818 Patentlaan 2<br>NL - 2280 HV Rijswijk<br>Tel. (+31-70) 340-2040, Tx. 31 651 epo nl,<br>Fax: (+31-70) 340-3016 | Augarde, E |

1

Form PCT/ISA/210 (second sheet) (July 1992)

Int onal Application No
PCT/IT 98/00056

C.(Continuation) DOCUMENTS CONSIDERED TO BE RELEVANT

| Category | Citation of document, with indication, where appropriate, of the relevant passages | Relevant to claim No. |
|---|---|---|
| A | YAMAUCHI H ET AL: "A 24X-SPEED CIRC DECODER FOR A CD-DSP/CD-ROM DECODER LSI" IEEE TRANSACTIONS ON CONSUMER ELECTRONICS, vol. 43, no. 3, August 1997, pages 483-490, XP002055400 see paragraph 4.7 see paragraph 5 ----- | 2 |

# INTERNATIONAL SEARCH REPORT

information on patent family members

Inte onal Application No
PCT/IT 98/00056

| Patent document cited in search report | Publication date | Patent family member(s) | Publication date |
|---|---|---|---|
| EP 0821493 A | 28-01-1900 | FR 2751810 A | 30-01-1998 |
| | | JP 10117148 A | 06-05-1998 |
| US 5691994 A | 25-11-1997 | NONE | |
| US 5631914 A | 20-05-1997 | JP 2029032 A | 31-01-1990 |
| | | JP 2039728 A | 08-02-1990 |
| | | JP 2774513 B | 09-07-1998 |
| | | US 5068855 A | 26-11-1991 |